pandas

September 18, 2021

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

[2]:
```
dict1={"Name":['Prateek','Rahul','Roshan','Abhishek'],
       "Marks":['98','45','76','64'],
       "City":['Varanasi','Delhi','Mumbai','Lucknow']}
```

[3]:
```
df=pd.DataFrame(dict1)
```

[4]:
```
df
```

[4]:

	Name	Marks	City
0	Prateek	98	Varanasi
1	Rahul	45	Delhi
2	Roshan	76	Mumbai
3	Abhishek	64	Lucknow

[5]:
```
df.to_csv('Friends.csv')
```

[6]:
```
df.to_csv('Friendsnoindex.csv', index=False)
```

[7]:
```
df.head()
```

[7]:

	Name	Marks	City
0	Prateek	98	Varanasi
1	Rahul	45	Delhi
2	Roshan	76	Mumbai
3	Abhishek	64	Lucknow

[8]:
```
df.tail()
```

[8]:

	Name	Marks	City
0	Prateek	98	Varanasi
1	Rahul	45	Delhi
2	Roshan	76	Mumbai
3	Abhishek	64	Lucknow

[9]:
```
df.describe()
```

[9]:
```
           Name Marks      City
count         4     4         4
unique        4     4         4
top     Prateek    98  Varanasi
freq          1     1         1
```

[10]:
```
df['Name']
```

[10]:
```
0     Prateek
1       Rahul
2      Roshan
3    Abhishek
Name: Name, dtype: object
```

[11]:
```
df['Name'][2]
```

[11]:
```
'Roshan'
```

[12]:
```
df['Name'][2]='Golesar'
```

[13]:
```
df
```

[13]:
```
       Name Marks      City
0   Prateek    98  Varanasi
1     Rahul    45     Delhi
2   Golesar    76    Mumbai
3  Abhishek    64   Lucknow
```

[14]:
```
df.to_csv('namechange.csv')
```

[15]:
```
df.index=['1st','2nd','3rd','4th']
```

[16]:
```
df
```

[16]:
```
         Name Marks      City
1st   Prateek    98  Varanasi
2nd     Rahul    45     Delhi
3rd   Golesar    76    Mumbai
4th  Abhishek    64   Lucknow
```

[17]:
```
df.to_csv('newownchanges', index=False)
```

[ ]:
```

```

[ ]:
```

```

[18]:
```
ser=pd.Series(np.random.rand(23))
```

[19]:
```
ser
```

[19]:
```
0     0.965262
1     0.485560
2     0.510211
3     0.849653
4     0.171309
5     0.616993
6     0.767631
7     0.282703
8     0.316169
9     0.805976
10    0.209138
11    0.258779
12    0.485600
13    0.144315
14    0.309638
15    0.775186
16    0.681616
17    0.923948
18    0.677623
19    0.020685
20    0.037263
21    0.650880
22    0.294196
dtype: float64
```

[20]:
```
type(ser)
```

[20]:
```
pandas.core.series.Series
```

[21]:
```
newdf=pd.DataFrame(np.random.rand(40,4), index=np.arange(40))
```

[22]:
```
newdf
```

[22]:

	0	1	2	3
0	0.552158	0.651795	0.973201	0.982302
1	0.092994	0.897207	0.323332	0.094031
2	0.428754	0.374083	0.055701	0.624566
3	0.971488	0.017363	0.121972	0.947491
4	0.254130	0.140490	0.497646	0.933547
5	0.954719	0.668797	0.478453	0.042606
6	0.122641	0.588573	0.951712	0.467526
7	0.358812	0.080251	0.948875	0.098566
8	0.400856	0.805794	0.362886	0.240607
9	0.004489	0.333342	0.260608	0.026544
10	0.251316	0.824403	0.576485	0.126396

```
11  0.514388  0.642302  0.625816  0.481233
12  0.194873  0.191091  0.343896  0.479190
13  0.775112  0.712145  0.594174  0.099057
14  0.099181  0.583111  0.790376  0.705131
15  0.161054  0.111730  0.992583  0.634086
16  0.767001  0.481431  0.012333  0.772586
17  0.776824  0.979741  0.936100  0.696165
18  0.035991  0.921601  0.713101  0.961465
19  0.158198  0.726272  0.307848  0.944272
20  0.395764  0.886345  0.801767  0.895417
21  0.873492  0.168100  0.940839  0.284848
22  0.621339  0.203030  0.099535  0.864084
23  0.394480  0.829471  0.442451  0.006871
24  0.601835  0.780288  0.758905  0.473584
25  0.250012  0.746560  0.044894  0.227086
26  0.667460  0.112693  0.343799  0.954969
27  0.665220  0.288364  0.418022  0.743268
28  0.450248  0.783526  0.583676  0.223974
29  0.989774  0.182877  0.234314  0.652587
30  0.186828  0.115991  0.550990  0.959070
31  0.835644  0.967993  0.557474  0.494073
32  0.190630  0.211081  0.217685  0.099548
33  0.936595  0.883744  0.872077  0.355385
34  0.844353  0.153619  0.615026  0.128763
35  0.387048  0.994715  0.218834  0.022107
36  0.840149  0.886846  0.758205  0.462533
37  0.459119  0.051516  0.393806  0.694245
38  0.475052  0.739866  0.175719  0.239396
39  0.655573  0.208966  0.342171  0.139032
```

[23]:
```
newdf.head()
```

[23]:
```
          0         1         2         3
0  0.552158  0.651795  0.973201  0.982302
1  0.092994  0.897207  0.323332  0.094031
2  0.428754  0.374083  0.055701  0.624566
3  0.971488  0.017363  0.121972  0.947491
4  0.254130  0.140490  0.497646  0.933547
```

[24]:
```
type(newdf)
```

[24]:
```
pandas.core.frame.DataFrame
```

[25]:
```
newdf.describe()
```

[25]:
```
                0          1          2          3
count  40.000000  40.000000  40.000000  40.000000
```

```
mean    0.489890   0.523178   0.505932   0.481955
std     0.294409   0.327399   0.291972   0.333977
min     0.004489   0.017363   0.012333   0.006871
25%     0.236227   0.189038   0.296038   0.136465
50%     0.454684   0.615437   0.488049   0.476387
75%     0.769029   0.810446   0.758380   0.750597
max     0.989774   0.994715   0.992583   0.982302
```

[26]:
```
newdf.dtypes
```

[26]:
```
0    float64
1    float64
2    float64
3    float64
dtype: object
```

[27]:
```
newdf[2][2]="Prateek"
```

[28]:
```
newdf.head()
```

[28]:

	0	1	2	3
0	0.552158	0.651795	0.973201	0.982302
1	0.092994	0.897207	0.323332	0.094031
2	0.428754	0.374083	Prateek	0.624566
3	0.971488	0.017363	0.121972	0.947491
4	0.254130	0.140490	0.497646	0.933547

[29]:
```
newdf.index
```

[29]:
```
Int64Index([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
            17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
            34, 35, 36, 37, 38, 39],
           dtype='int64')
```

[30]:
```
newdf.columns
```

[30]:
```
RangeIndex(start=0, stop=4, step=1)
```

[31]:
```
newdf.to_numpy()
```

[31]:
```
array([[0.5521577329503847, 0.6517953175920891, 0.9732014909832583,
        0.9823019838501279],
       [0.09299448626468132, 0.8972068269567478, 0.32333234484596163,
        0.09403059125779667],
       [0.4287543997867266, 0.37408288751621877, 'Prateek',
        0.6245658040368015],
       [0.9714880382697068, 0.017363374111682983, 0.12197195946863759,
        0.9474905749139856],
```

```
[0.25413045214052765, 0.14048968444460086, 0.4976459730245877,
 0.9335466095652171],
[0.9547190741810992, 0.6687970756389736, 0.4784527603444457,
 0.0426061849697259],
[0.12264102131136334, 0.5885727230093342, 0.9517119579459441,
 0.4675262117395136],
[0.35881248439189317, 0.08025104382740711, 0.9488750650321868,
 0.0985664970860981],
[0.4008564914378858, 0.8057935060411565, 0.3628859645022805,
 0.2406074980822449],
[0.004488534022146995, 0.3333415598235241, 0.26060754085532656,
 0.02654418923866697],
[0.2513160475789281, 0.8244026343805458, 0.5764851798606966,
 0.12639626972564866],
[0.5143883406174391, 0.6423017041112622, 0.6258163428200616,
 0.4812332252510808],
[0.1948726287871747, 0.19109108500655136, 0.34389578915060814,
 0.4791903033288505],
[0.7751124045805189, 0.712145118922811, 0.594174373431228,
 0.09905706408958403],
[0.09918065062172554, 0.5831110391684372, 0.7903759444478913,
 0.705131054469735],
[0.16105417024177626, 0.11173019597641598, 0.9925829471903014,
 0.6340857771837121],
[0.7670005992355582, 0.48143126927243174, 0.01233325827969578,
 0.7725859168461018],
[0.7768242697090615, 0.9797411580628915, 0.9360999733895013,
 0.6961648903239759],
[0.03599071900944728, 0.9216008236554968, 0.7131014364261895,
 0.9614651621361425],
[0.1581979033818115, 0.7262723827258715, 0.3078477121629476,
 0.944272258900233],
[0.3957644590981204, 0.8863446194301231, 0.801766832520428,
 0.8954171002489877],
[0.8734920657370713, 0.16810021058958868, 0.9408385063526261,
 0.28484778190872073],
[0.6213390117949082, 0.203030200005464, 0.09953491446442808,
 0.8640835723239045],
[0.3944797641669687, 0.8294707830732762, 0.4424508651423754,
 0.006871297437194279],
[0.6018351923004939, 0.7802881888358025, 0.758904991436731,
 0.47358407046729634],
[0.25001200189541795, 0.7465602938482847, 0.044894108971450075,
 0.22708560344419182],
[0.6674600536109091, 0.11269318371208814, 0.343798947230322,
 0.9549690642567558],
[0.6652203985977215, 0.2883639063711434, 0.41802215976289203,
```

```
 0.7432677920212395],
[0.45024793328406254, 0.7835255167954654, 0.5836764384534493,
 0.22397392438711283],
[0.9897738306788139, 0.18287678391508655, 0.23431445810383322,
 0.6525871635305716],
[0.1868281531597341, 0.11599092451390647, 0.5509904952146276,
 0.9590701630190231],
[0.8356442545317192, 0.9679925436315724, 0.5574744058158707,
 0.4940734488504528],
[0.19062997775562518, 0.21108096390256004, 0.21768492082063184,
 0.0995480952015041],
[0.9365954281105492, 0.8837438849792439, 0.872077379000511,
 0.3553854778395573],
[0.8443534870730202, 0.1536187779477054, 0.6150256224582743,
 0.12876267927874963],
[0.3870480764765486, 0.9947149920143641, 0.2188338391017025,
 0.02210658366530671],
[0.8401491354709761, 0.8868463527169632, 0.7582047949352969,
 0.4625333696805157],
[0.45911944600553767, 0.05151559451854837, 0.39380624978121215,
 0.6942449013732291],
[0.475052015029909, 0.7398657573710093, 0.17571856952439202,
 0.23939646763034172],
[0.6555726372121937, 0.20896620118262, 0.3421712313594796,
 0.1390317844527078]], dtype=object)
```

[32]:
```
newdf.head()
```

[32]:

	0	1	2	3
0	0.552158	0.651795	0.973201	0.982302
1	0.092994	0.897207	0.323332	0.094031
2	0.428754	0.374083	Prateek	0.624566
3	0.971488	0.017363	0.121972	0.947491
4	0.254130	0.140490	0.497646	0.933547

[33]:
```
newdf[2][2]='34'
```

[34]:
```
newdf.head()
```

[34]:

	0	1	2	3
0	0.552158	0.651795	0.973201	0.982302
1	0.092994	0.897207	0.323332	0.094031
2	0.428754	0.374083	34	0.624566
3	0.971488	0.017363	0.121972	0.947491
4	0.254130	0.140490	0.497646	0.933547

[35]:
```
newdf.T.head()
```

[35]:
```
          0         1         2         3         4         5         6   \
0  0.552158  0.092994  0.428754  0.971488   0.25413  0.954719  0.122641
1  0.651795  0.897207  0.374083  0.017363   0.14049  0.668797  0.588573
2  0.973201  0.323332        34  0.121972  0.497646  0.478453  0.951712
3  0.982302  0.094031  0.624566  0.947491  0.933547  0.042606  0.467526

          7         8         9   …        30        31        32        33  \
0  0.358812  0.400856  0.004489  …  0.186828  0.835644   0.19063  0.936595
1  0.080251  0.805794  0.333342  …  0.115991  0.967993  0.211081  0.883744
2  0.948875  0.362886  0.260608  …   0.55099  0.557474  0.217685  0.872077
3  0.098566  0.240607  0.026544  …   0.95907  0.494073  0.099548  0.355385

          34        35        36        37        38        39
0  0.844353  0.387048  0.840149  0.459119  0.475052  0.655573
1  0.153619  0.994715  0.886846  0.051516  0.739866  0.208966
2  0.615026  0.218834  0.758205  0.393806  0.175719  0.342171
3  0.128763  0.022107  0.462533  0.694245  0.239396  0.139032

[4 rows x 40 columns]
```

[36]:
```
newdf.sort_index(axis=0, ascending=False) #axis=0 means row
```

[36]:
```
           0         1         2         3
39  0.655573  0.208966  0.342171  0.139032
38  0.475052  0.739866  0.175719  0.239396
37  0.459119  0.051516  0.393806  0.694245
36  0.840149  0.886846  0.758205  0.462533
35  0.387048  0.994715  0.218834  0.022107
34  0.844353  0.153619  0.615026  0.128763
33  0.936595  0.883744  0.872077  0.355385
32  0.190630  0.211081  0.217685  0.099548
31  0.835644  0.967993  0.557474  0.494073
30  0.186828  0.115991   0.55099  0.959070
29  0.989774  0.182877  0.234314  0.652587
28  0.450248  0.783526  0.583676  0.223974
27  0.665220  0.288364  0.418022  0.743268
26  0.667460  0.112693  0.343799  0.954969
25  0.250012  0.746560  0.044894  0.227086
24  0.601835  0.780288  0.758905  0.473584
23  0.394480  0.829471  0.442451  0.006871
22  0.621339  0.203030  0.099535  0.864084
21  0.873492  0.168100  0.940839  0.284848
20  0.395764  0.886345  0.801767  0.895417
19  0.158198  0.726272  0.307848  0.944272
18  0.035991  0.921601  0.713101  0.961465
17  0.776824  0.979741    0.9361  0.696165
16  0.767001  0.481431  0.012333  0.772586
```

```
15  0.161054  0.111730  0.992583  0.634086
14  0.099181  0.583111  0.790376  0.705131
13  0.775112  0.712145  0.594174  0.099057
12  0.194873  0.191091  0.343896  0.479190
11  0.514388  0.642302  0.625816  0.481233
10  0.251316  0.824403  0.576485  0.126396
9   0.004489  0.333342  0.260608  0.026544
8   0.400856  0.805794  0.362886  0.240607
7   0.358812  0.080251  0.948875  0.098566
6   0.122641  0.588573  0.951712  0.467526
5   0.954719  0.668797  0.478453  0.042606
4   0.254130  0.140490  0.497646  0.933547
3   0.971488  0.017363  0.121972  0.947491
2   0.428754  0.374083        34  0.624566
1   0.092994  0.897207  0.323332  0.094031
0   0.552158  0.651795  0.973201  0.982302
```

[37]:
```
newdf.head()
```

[37]:
```
          0         1         2         3
0  0.552158  0.651795  0.973201  0.982302
1  0.092994  0.897207  0.323332  0.094031
2  0.428754  0.374083        34  0.624566
3  0.971488  0.017363  0.121972  0.947491
4  0.254130  0.140490  0.497646  0.933547
```

[38]:
```
newdf[0]
```

[38]:
```
0     0.552158
1     0.092994
2     0.428754
3     0.971488
4     0.254130
5     0.954719
6     0.122641
7     0.358812
8     0.400856
9     0.004489
10    0.251316
11    0.514388
12    0.194873
13    0.775112
14    0.099181
15    0.161054
16    0.767001
17    0.776824
18    0.035991
```

```
19    0.158198
20    0.395764
21    0.873492
22    0.621339
23    0.394480
24    0.601835
25    0.250012
26    0.667460
27    0.665220
28    0.450248
29    0.989774
30    0.186828
31    0.835644
32    0.190630
33    0.936595
34    0.844353
35    0.387048
36    0.840149
37    0.459119
38    0.475052
39    0.655573
Name: 0, dtype: float64
```

[39]:
```
type(newdf[0])
```

[39]:
```
pandas.core.series.Series
```

[40]:
```
#dont do newdf2=newdf like statement otherwise if you change the newdf2 then it␣
 ↪automatically change the newdf
#if you want to do this then do this
newdf2=newdf.copy()
```

[41]:
```
newdf2.head()
```

[41]:

	0	1	2	3
0	0.552158	0.651795	0.973201	0.982302
1	0.092994	0.897207	0.323332	0.094031
2	0.428754	0.374083	34	0.624566
3	0.971488	0.017363	0.121972	0.947491
4	0.254130	0.140490	0.497646	0.933547

[42]:
```
newdf.columns=['a','b','c','d'] #you can also do newdf.columns(list("abcd"))
```

[43]:
```
newdf.head(1)
```

[43]:

	a	b	c	d
0	0.552158	0.651795	0.973201	0.982302

[44]:
```
newdf.loc[0,0]=1
```

[45]:
```
newdf.head()
```

[45]:

	a	b	c	d	0
0	0.552158	0.651795	0.973201	0.982302	1.0
1	0.092994	0.897207	0.323332	0.094031	NaN
2	0.428754	0.374083	34	0.624566	NaN
3	0.971488	0.017363	0.121972	0.947491	NaN
4	0.254130	0.140490	0.497646	0.933547	NaN

[46]:
```
newdf.loc[0,'a']
```

[46]:
```
0.5521577329503847
```

[47]:
```
newdf=newdf.drop(0, axis=1).head()
```

[48]:
```
newdf.head()
```

[48]:

	a	b	c	d
0	0.552158	0.651795	0.973201	0.982302
1	0.092994	0.897207	0.323332	0.094031
2	0.428754	0.374083	34	0.624566
3	0.971488	0.017363	0.121972	0.947491
4	0.254130	0.140490	0.497646	0.933547

[49]:
```
newdf.loc[[3,4],['b','c']]
```

[49]:

	b	c
3	0.017363	0.121972
4	0.140490	0.497646

[50]:
```
newdf.loc[[3,4],:]
```

[50]:

	a	b	c	d
3	0.971488	0.017363	0.121972	0.947491
4	0.254130	0.140490	0.497646	0.933547

[51]:
```
newdf.loc[newdf['a']<0.9]
```

[51]:

	a	b	c	d
0	0.552158	0.651795	0.973201	0.982302
1	0.092994	0.897207	0.323332	0.094031
2	0.428754	0.374083	34	0.624566
4	0.254130	0.140490	0.497646	0.933547

[52]:
```
newdf.loc[(newdf['a']<0.9) & ( newdf['d']>0.2)]
```

[52]:
```
          a         b         c         d
0  0.552158  0.651795  0.973201  0.982302
2  0.428754  0.374083        34  0.624566
4  0.254130  0.140490  0.497646  0.933547
```

[53]:
```
newdf.loc[0,'c']
```

[53]:
```
0.9732014909832583
```

[54]:
```
newdf.iloc[1,1]
```

[54]:
```
0.8972068269567478
```

[55]:
```
newdf.drop(['a','c'], axis=1) #but if you use statement newdf.drop(['a','c'],␣
 ↪axis=1 , inplace=True) then it also chages
# the original dataframe
```

[55]:
```
          b         d
0  0.651795  0.982302
1  0.897207  0.094031
2  0.374083  0.624566
3  0.017363  0.947491
4  0.140490  0.933547
```

[56]:
```
newdf.head()
```

[56]:
```
          a         b         c         d
0  0.552158  0.651795  0.973201  0.982302
1  0.092994  0.897207  0.323332  0.094031
2  0.428754  0.374083        34  0.624566
3  0.971488  0.017363  0.121972  0.947491
4  0.254130  0.140490  0.497646  0.933547
```

[57]:
```
newdf.reset_index(drop=True, inplace=True)
```

[58]:
```
newdf
```

[58]:
```
          a         b         c         d
0  0.552158  0.651795  0.973201  0.982302
1  0.092994  0.897207  0.323332  0.094031
2  0.428754  0.374083        34  0.624566
3  0.971488  0.017363  0.121972  0.947491
4  0.254130  0.140490  0.497646  0.933547
```

[59]:
```
newdff=pd.DataFrame(np.random.rand(40,4), index=np.arange(40))
```

[60]:
```
newdff.head()
```

[60]:
```
          0         1         2         3
0  0.370044  0.493149  0.579273  0.674443
1  0.973562  0.314019  0.261455  0.219151
2  0.328934  0.680264  0.970098  0.180507
3  0.565297  0.356986  0.577119  0.012234
4  0.566876  0.764359  0.949248  0.910997
```

[61]:
```
newdff.head()
```

[61]:
```
          0         1         2         3
0  0.370044  0.493149  0.579273  0.674443
1  0.973562  0.314019  0.261455  0.219151
2  0.328934  0.680264  0.970098  0.180507
3  0.565297  0.356986  0.577119  0.012234
4  0.566876  0.764359  0.949248  0.910997
```

[62]:
```
newdf.head()
```

[62]:
```
          a         b         c         d
0  0.552158  0.651795  0.973201  0.982302
1  0.092994  0.897207  0.323332  0.094031
2  0.428754  0.374083        34  0.624566
3  0.971488  0.017363  0.121972  0.947491
4  0.254130  0.140490  0.497646  0.933547
```

[63]:
```
newdf.columns=[0,1,2,3]
```

[64]:
```
newdf.head()
```

[64]:
```
          0         1         2         3
0  0.552158  0.651795  0.973201  0.982302
1  0.092994  0.897207  0.323332  0.094031
2  0.428754  0.374083        34  0.624566
3  0.971488  0.017363  0.121972  0.947491
4  0.254130  0.140490  0.497646  0.933547
```

[65]:
```
newdff.head()
```

[65]:
```
          0         1         2         3
0  0.370044  0.493149  0.579273  0.674443
1  0.973562  0.314019  0.261455  0.219151
2  0.328934  0.680264  0.970098  0.180507
3  0.565297  0.356986  0.577119  0.012234
4  0.566876  0.764359  0.949248  0.910997
```

[66]:
```
con=pd.concat([newdf, newdff])
```

[67]:
```
con
```

[67]:

	0	1	2	3
0	0.552158	0.651795	0.973201	0.982302
1	0.092994	0.897207	0.323332	0.094031
2	0.428754	0.374083	34	0.624566
3	0.971488	0.017363	0.121972	0.947491
4	0.254130	0.140490	0.497646	0.933547
0	0.370044	0.493149	0.579273	0.674443
1	0.973562	0.314019	0.261455	0.219151
2	0.328934	0.680264	0.970098	0.180507
3	0.565297	0.356986	0.577119	0.012234
4	0.566876	0.764359	0.949248	0.910997
5	0.193423	0.425836	0.026817	0.149899
6	0.909758	0.706887	0.099913	0.654798
7	0.069718	0.118955	0.359878	0.865380
8	0.560532	0.255943	0.650184	0.850571
9	0.631892	0.843851	0.613083	0.987060
10	0.617083	0.968555	0.89976	0.572864
11	0.075249	0.940214	0.291114	0.966133
12	0.112729	0.075980	0.984677	0.249643
13	0.207094	0.790369	0.215284	0.857654
14	0.503223	0.537988	0.167501	0.913833
15	0.251782	0.571860	0.411965	0.717614
16	0.797887	0.788813	0.336416	0.887048
17	0.242866	0.117050	0.941977	0.253344
18	0.281973	0.111493	0.730959	0.833585
19	0.129621	0.828636	0.208489	0.270576
20	0.655524	0.484194	0.068602	0.089233
21	0.988779	0.647268	0.434784	0.375075
22	0.003599	0.110532	0.509118	0.174459
23	0.921166	0.237965	0.573482	0.161625
24	0.192809	0.350684	0.135056	0.556007
25	0.232991	0.472614	0.612762	0.954406
26	0.861417	0.762200	0.899865	0.233715
27	0.598403	0.189501	0.357842	0.312689
28	0.479379	0.286507	0.218367	0.786236
29	0.991960	0.088184	0.041418	0.775719
30	0.764625	0.841654	0.328212	0.364446
31	0.899505	0.988533	0.134854	0.783869
32	0.560092	0.011740	0.934825	0.313373
33	0.716579	0.838667	0.666672	0.624261
34	0.272800	0.310084	0.447957	0.252140
35	0.786200	0.327248	0.661574	0.664496
36	0.094790	0.465710	0.064407	0.694679
37	0.734640	0.318369	0.111998	0.135189
38	0.915842	0.095534	0.175264	0.799705
39	0.668275	0.690968	0.423437	0.222168

[68]:
```
dff= pd.DataFrame({"name": ['NaN', 'Batman', 'Catwoman'],
                   "toy": [np.nan, 'Batmobile', 'Bullwhip'],
                   "born": [pd.NaT, pd.Timestamp("1940-04-25"),
                            pd.NaT]})
```

[69]:
```
dff.to_csv('dff1', index=False)
```

[70]:
```
dff1=pd.read_csv('dff1')
```

[71]:
```
dff1
```

[71]:

	name	toy	born
0	NaN	NaN	NaN
1	Batman	Batmobile	1940-04-25
2	Catwoman	Bullwhip	NaN

[72]:
```
dff1['name'].isnull
```

[72]:
```
<bound method Series.isnull of 0         NaN
1      Batman
2    Catwoman
Name: name, dtype: object>
```

[73]:
```
dff1.dropna(how='all')
```

[73]:

	name	toy	born
1	Batman	Batmobile	1940-04-25
2	Catwoman	Bullwhip	NaN

[ ]:

[ ]: