CS122APandasDataframes

November 23, 2021

1 Dataframes: Another view of tabular data!

This notebook introduces the basics of Pandas and dataframes from a relational/SQL perspective. A Pandas dataframe is essentially a collection of numpy series that share an index. Dataframes can be populated with data from CSV files, JSON files, Parquet files, SQL databases, and other sources.

[40]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

[41]:
```
# first let's populate a dataframe with sailor data from a CSV file

sailors = pd.read_csv("/Users/mikejcarey/SailingClub/Sailors.csv")
```

[42]:
```
cat /Users/mikejcarey/SailingClub/Sailors.csv
```

```
sid,sname,rating,age,major
22,Dustin,7,45,CS
29,Brutus,1,33,EE
31,Lubber,8,55.5,Econ
32,Andy,8,25.5,Math
58,Rusty,10,35,CS
64,Horatio,7,35,CS
71,Zorba,10,16,CS
74,Horatio,9,35,Math
85,Art,4,25.5,Music
95,Bob,3,63.5,Econ
101,Joan,3,,Math
107,Johannes,,35.0,
```

[43]:
```
sailors
```

[43]:

	sid	sname	rating	age	major
0	22	Dustin	7.0	45.0	CS
1	29	Brutus	1.0	33.0	EE
2	31	Lubber	8.0	55.5	Econ
3	32	Andy	8.0	25.5	Math

```
4    58     Rusty    10.0  35.0     CS
5    64   Horatio     7.0  35.0     CS
6    71     Zorba    10.0  16.0     CS
7    74   Horatio     9.0  35.0   Math
8    85       Art     4.0  25.5  Music
9    95       Bob     3.0  63.5   Econ
10  101      Joan     3.0   NaN   Math
11  107  Johannes     NaN  35.0    NaN
```

[44]:
```
sailors.dtypes
```

[44]:
```
sid         int64
sname      object
rating    float64
age       float64
major      object
dtype: object
```

[45]:
```
sailors.describe()
```

[45]:
```
              sid     rating        age
count   12.000000  11.000000  11.000000
mean    64.083333   6.363636  36.727273
std     30.010478   3.107176  13.583747
min     22.000000   1.000000  16.000000
25%     31.750000   3.500000  29.250000
50%     67.500000   7.000000  35.000000
75%     87.500000   8.500000  40.000000
max    107.000000  10.000000  63.500000
```

[46]:
```
# now let's populate two more dataframes with boat and reservation data

boats = pd.read_csv("/Users/mikejcarey/SailingClub/Boats.csv")
reserves = pd.read_csv("/Users/mikejcarey/SailingClub/Reserves.csv")
```

[47]:
```
boats
```

[47]:
```
   bid      bname  color
0  101  Interlake   blue
1  102  Interlake    red
2  103    Clipper  green
3  104     Marine    red
```

[48]:
```
reserves
```

[48]:
```
     sid    bid        date
0   22.0  101.0  1998-10-10
```

```
1   22.0  102.0  1998-10-10
2   22.0  103.0  1998-10-08
3   22.0  104.0  1998-10-07
4   31.0  102.0  1998-10-11
5   31.0  103.0  1998-11-06
6   31.0  104.0  1998-11-12
7   64.0  101.0  1998-09-05
8   64.0  102.0  1998-09-02
9   74.0  103.0  1993-09-08
10   NaN  103.0  1998-09-09
11   1.0    NaN  2001-01-11
12   1.0    NaN  2002-02-02
```

[49]:
```
# now let's work with these three dataframes
sailors
```

[49]:

	sid	sname	rating	age	major
0	22	Dustin	7.0	45.0	CS
1	29	Brutus	1.0	33.0	EE
2	31	Lubber	8.0	55.5	Econ
3	32	Andy	8.0	25.5	Math
4	58	Rusty	10.0	35.0	CS
5	64	Horatio	7.0	35.0	CS
6	71	Zorba	10.0	16.0	CS
7	74	Horatio	9.0	35.0	Math
8	85	Art	4.0	25.5	Music
9	95	Bob	3.0	63.5	Econ
10	101	Joan	3.0	NaN	Math
11	107	Johannes	NaN	35.0	NaN

[50]:
```
sailors.head(3)
```

[50]:

	sid	sname	rating	age	major
0	22	Dustin	7.0	45.0	CS
1	29	Brutus	1.0	33.0	EE
2	31	Lubber	8.0	55.5	Econ

[51]:
```
sailors.tail(2)
```

[51]:

	sid	sname	rating	age	major
10	101	Joan	3.0	NaN	Math
11	107	Johannes	NaN	35.0	NaN

[52]:
```
# pandas dataframes are ordered and indexed -- unlike relations
# in the pure relational model or tables in SQL!

sailors[10:12]
```

[52]:
```
     sid     sname  rating   age major
10   101      Joan     3.0   NaN  Math
11   107  Johannes     NaN  35.0   NaN
```

[53]:
```
# we can do relational-style projection

sailors['sname']
```

[53]:
```
0       Dustin
1       Brutus
2       Lubber
3         Andy
4        Rusty
5      Horatio
6        Zorba
7      Horatio
8          Art
9          Bob
10        Joan
11    Johannes
Name: sname, dtype: object
```

[54]:
```
sailors[['sname','rating']]
```

[54]:
```
        sname  rating
0     Dustin     7.0
1     Brutus     1.0
2     Lubber     8.0
3       Andy     8.0
4      Rusty    10.0
5    Horatio     7.0
6      Zorba    10.0
7    Horatio     9.0
8        Art     4.0
9        Bob     3.0
10      Joan     3.0
11  Johannes     NaN
```

[55]:
```
# we can do relational-style selection

sailors[sailors.age < 30.0]
```

[55]:
```
    sid  sname  rating   age  major
3    32   Andy     8.0  25.5   Math
6    71  Zorba    10.0  16.0     CS
8    85    Art     4.0  25.5  Music
```

[58]:
```
# let's break this down - first of all, each
# column of a dataframe is an indexed serles

sailors.age
```

[58]:
```
0     45.0
1     33.0
2     55.5
3     25.5
4     35.0
5     35.0
6     16.0
7     35.0
8     25.5
9     63.5
10     NaN
11    35.0
Name: age, dtype: float64
```

[59]:
```
sailors.age < 30.0
```

[59]:
```
0     False
1     False
2     False
3      True
4     False
5     False
6      True
7     False
8      True
9     False
10    False
11    False
Name: age, dtype: bool
```

[60]:
```
sailors[sailors.age < 30.0]
```

[60]:

	sid	sname	rating	age	major
3	32	Andy	8.0	25.5	Math
6	71	Zorba	10.0	16.0	CS
8	85	Art	4.0	25.5	Music

[61]:
```
sailors[(sailors.major == 'CS') | (sailors.major == 'EE')]
```

[61]:

	sid	sname	rating	age	major
0	22	Dustin	7.0	45.0	CS
1	29	Brutus	1.0	33.0	EE

```
4   58    Rusty    10.0  35.0    CS
5   64  Horatio     7.0  35.0    CS
6   71    Zorba    10.0  16.0    CS
```

[62]:
```
sailors[(sailors.major == 'CS') & (sailors.rating >= 8.0)]
```

[62]:
```
   sid  sname  rating   age major
4   58  Rusty    10.0  35.0    CS
6   71  Zorba    10.0  16.0    CS
```

[64]:
```
sailors
```

[64]:
```
     sid     sname  rating   age  major
0    22    Dustin     7.0  45.0     CS
1    29    Brutus     1.0  33.0     EE
2    31    Lubber     8.0  55.5   Econ
3    32      Andy     8.0  25.5   Math
4    58     Rusty    10.0  35.0     CS
5    64   Horatio     7.0  35.0     CS
6    71     Zorba    10.0  16.0     CS
7    74   Horatio     9.0  35.0   Math
8    85       Art     4.0  25.5  Music
9    95       Bob     3.0  63.5   Econ
10  101      Joan     3.0   NaN   Math
11  107  Johannes     NaN  35.0    NaN
```

[63]:
```
# and one more time, let's break this down step-by-step

(sailors.major == 'CS')
```

[63]:
```
0      True
1     False
2     False
3     False
4      True
5      True
6      True
7     False
8     False
9     False
10    False
11    False
Name: major, dtype: bool
```

[65]:
```
(sailors.rating >= 8.0)
```

[65]:
```
0     False
1     False
2      True
3      True
4      True
5     False
6      True
7      True
8     False
9     False
10    False
11    False
Name: rating, dtype: bool
```

[66]:
```
(sailors.major == 'CS') & (sailors.rating >= 8.0)
```

[66]:
```
0     False
1     False
2     False
3     False
4      True
5     False
6      True
7     False
8     False
9     False
10    False
11    False
dtype: bool
```

[67]:
```
sailors[(sailors.major == 'CS') & (sailors.rating >= 8.0)]
```

[67]:

	sid	sname	rating	age	major
4	58	Rusty	10.0	35.0	CS
6	71	Zorba	10.0	16.0	CS

[68]:
```
# now let's do some multi-dataframe operations

boats
```

[68]:

	bid	bname	color
0	101	Interlake	blue
1	102	Interlake	red
2	103	Clipper	green
3	104	Marine	red

[69]:
```
reserves
```

[69]:

	sid	bid	date
0	22.0	101.0	1998-10-10
1	22.0	102.0	1998-10-10
2	22.0	103.0	1998-10-08
3	22.0	104.0	1998-10-07
4	31.0	102.0	1998-10-11
5	31.0	103.0	1998-11-06
6	31.0	104.0	1998-11-12
7	64.0	101.0	1998-09-05
8	64.0	102.0	1998-09-02
9	74.0	103.0	1993-09-08
10	NaN	103.0	1998-09-09
11	1.0	NaN	2001-01-11
12	1.0	NaN	2002-02-02

[70]:
```
# we can do relational-style joins

pd.merge(reserves, boats, on='bid')
```

[70]:

	sid	bid	date	bname	color
0	22.0	101.0	1998-10-10	Interlake	blue
1	64.0	101.0	1998-09-05	Interlake	blue
2	22.0	102.0	1998-10-10	Interlake	red
3	31.0	102.0	1998-10-11	Interlake	red
4	64.0	102.0	1998-09-02	Interlake	red
5	22.0	103.0	1998-10-08	Clipper	green
6	31.0	103.0	1998-11-06	Clipper	green
7	74.0	103.0	1993-09-08	Clipper	green
8	NaN	103.0	1998-09-09	Clipper	green
9	22.0	104.0	1998-10-07	Marine	red
10	31.0	104.0	1998-11-12	Marine	red

[71]:
```
pd.merge(reserves, boats, left_on='bid', right_on='bid')
```

[71]:

	sid	bid	date	bname	color
0	22.0	101.0	1998-10-10	Interlake	blue
1	64.0	101.0	1998-09-05	Interlake	blue
2	22.0	102.0	1998-10-10	Interlake	red
3	31.0	102.0	1998-10-11	Interlake	red
4	64.0	102.0	1998-09-02	Interlake	red
5	22.0	103.0	1998-10-08	Clipper	green
6	31.0	103.0	1998-11-06	Clipper	green
7	74.0	103.0	1993-09-08	Clipper	green
8	NaN	103.0	1998-09-09	Clipper	green
9	22.0	104.0	1998-10-07	Marine	red
10	31.0	104.0	1998-11-12	Marine	red

[72]:
```
# relational-style joins can be inner joins (the default)
# or outer joins (left, right, outer)

pd.merge(sailors, reserves, on='sid', how='left')
```

[72]:

	sid	sname	rating	age	major	bid	date
0	22	Dustin	7.0	45.0	CS	101.0	1998-10-10
1	22	Dustin	7.0	45.0	CS	102.0	1998-10-10
2	22	Dustin	7.0	45.0	CS	103.0	1998-10-08
3	22	Dustin	7.0	45.0	CS	104.0	1998-10-07
4	29	Brutus	1.0	33.0	EE	NaN	NaN
5	31	Lubber	8.0	55.5	Econ	102.0	1998-10-11
6	31	Lubber	8.0	55.5	Econ	103.0	1998-11-06
7	31	Lubber	8.0	55.5	Econ	104.0	1998-11-12
8	32	Andy	8.0	25.5	Math	NaN	NaN
9	58	Rusty	10.0	35.0	CS	NaN	NaN
10	64	Horatio	7.0	35.0	CS	101.0	1998-09-05
11	64	Horatio	7.0	35.0	CS	102.0	1998-09-02
12	71	Zorba	10.0	16.0	CS	NaN	NaN
13	74	Horatio	9.0	35.0	Math	103.0	1993-09-08
14	85	Art	4.0	25.5	Music	NaN	NaN
15	95	Bob	3.0	63.5	Econ	NaN	NaN
16	101	Joan	3.0	NaN	Math	NaN	NaN
17	107	Johannes	NaN	35.0	NaN	NaN	NaN

[73]:
```
pd.merge(sailors, reserves, left_on='sid', right_on='sid', how='outer')
```

[73]:

	sid	sname	rating	age	major	bid	date
0	22.0	Dustin	7.0	45.0	CS	101.0	1998-10-10
1	22.0	Dustin	7.0	45.0	CS	102.0	1998-10-10
2	22.0	Dustin	7.0	45.0	CS	103.0	1998-10-08
3	22.0	Dustin	7.0	45.0	CS	104.0	1998-10-07
4	29.0	Brutus	1.0	33.0	EE	NaN	NaN
5	31.0	Lubber	8.0	55.5	Econ	102.0	1998-10-11
6	31.0	Lubber	8.0	55.5	Econ	103.0	1998-11-06
7	31.0	Lubber	8.0	55.5	Econ	104.0	1998-11-12
8	32.0	Andy	8.0	25.5	Math	NaN	NaN
9	58.0	Rusty	10.0	35.0	CS	NaN	NaN
10	64.0	Horatio	7.0	35.0	CS	101.0	1998-09-05
11	64.0	Horatio	7.0	35.0	CS	102.0	1998-09-02
12	71.0	Zorba	10.0	16.0	CS	NaN	NaN
13	74.0	Horatio	9.0	35.0	Math	103.0	1993-09-08
14	85.0	Art	4.0	25.5	Music	NaN	NaN
15	95.0	Bob	3.0	63.5	Econ	NaN	NaN
16	101.0	Joan	3.0	NaN	Math	NaN	NaN
17	107.0	Johannes	NaN	35.0	NaN	NaN	NaN
18	NaN	NaN	NaN	NaN	NaN	103.0	1998-09-09

```
19    1.0       NaN     NaN   NaN    NaN    NaN  2001-01-11
20    1.0       NaN     NaN   NaN    NaN    NaN  2002-02-02
```

[75]:
```
# we can also do relational-style unions (and more) - but first
# let's get some additional data, this time from a JSON file

moreboats = pd.read_json("/Users/mikejcarey/SailingClub/MoreBoats.json",␣
 ↪lines='true')
```

[76]:
```
boats
```

[76]:
```
   bid      bname  color
0  101  Interlake   blue
1  102  Interlake    red
2  103    Clipper  green
3  104     Marine    red
```

[77]:
```
moreboats
```

[77]:
```
   bid    bname  color
0  201  Sunfish   blue
1  202  Sunfish    red
2  203    Yacht  green
3  204    Barge    red
```

[78]:
```
pd.concat([boats, moreboats])
```

[78]:
```
   bid      bname  color
0  101  Interlake   blue
1  102  Interlake    red
2  103    Clipper  green
3  104     Marine    red
0  201    Sunfish   blue
1  202    Sunfish    red
2  203      Yacht  green
3  204      Barge    red
```

[79]:
```
# to do a more relational-style union (actually union all) we
# should fix the rather odd indexing that we got above (:-))

pd.concat([boats, moreboats], ignore_index=True)
```

[79]:
```
   bid      bname  color
0  101  Interlake   blue
1  102  Interlake    red
2  103    Clipper  green
3  104     Marine    red
```

```
4  201    Sunfish   blue
5  202    Sunfish    red
6  203      Yacht  green
7  204      Barge    red
```

[80]:
```
# we can also do relational-style grouping and aggregation (and more)

bymajor = sailors.groupby('major')
bymajor
```

[80]:
```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f8c20536e50>
```

[81]:
```
bymajor.size()
```

[81]:
```
major
CS       4
EE       1
Econ     2
Math     3
Music    1
dtype: int64
```

[82]:
```
bymajor.count()
```

[82]:
```
        sid  sname  rating  age
major
CS       4      4       4    4
EE       1      1       1    1
Econ     2      2       2    2
Math     3      3       3    2
Music    1      1       1    1
```

[84]:
```
bymajor.rating.max()
```

[84]:
```
major
CS       10.0
EE        1.0
Econ      8.0
Math      9.0
Music     4.0
Name: rating, dtype: float64
```

[85]:
```
bymajor.max()[['rating','age']]
```

[85]:
```
        rating   age
major
CS       10.0  45.0
```

```
EE        1.0  33.0
Econ      8.0  63.5
Math      9.0  35.0
Music     4.0  25.5
```

[86]:
```
bymajor.agg({'rating' : ['min', 'max'], 'age' : ['count','mean','std']})
```

[86]:
```
       rating         age
          min   max count   mean        std
major
CS        7.0  10.0     4  32.75  12.120919
EE        1.0   1.0     1  33.00        NaN
Econ      3.0   8.0     2  59.50   5.656854
Math      3.0   9.0     2  30.25   6.717514
Music     4.0   4.0     1  25.50        NaN
```

[87]:
```
# we can do relational-style ordering and limiting as well

bymajor.rating.max().sort_values(ascending=False)[:3]
```

[87]:
```
major
CS      10.0
Math     9.0
Econ     8.0
Name: rating, dtype: float64
```

[88]:
```
# we can also extend a dataframe with a new column, e.g., dogage

sailors.age / 7
```

[88]:
```
0     6.428571
1     4.714286
2     7.928571
3     3.642857
4     5.000000
5     5.000000
6     2.285714
7     5.000000
8     3.642857
9     9.071429
10         NaN
11    5.000000
Name: age, dtype: float64
```

[89]:
```
sailors.assign(dogage = sailors.age / 7)
```

[89]:

	sid	sname	rating	age	major	dogage
0	22	Dustin	7.0	45.0	CS	6.428571
1	29	Brutus	1.0	33.0	EE	4.714286
2	31	Lubber	8.0	55.5	Econ	7.928571
3	32	Andy	8.0	25.5	Math	3.642857
4	58	Rusty	10.0	35.0	CS	5.000000
5	64	Horatio	7.0	35.0	CS	5.000000
6	71	Zorba	10.0	16.0	CS	2.285714
7	74	Horatio	9.0	35.0	Math	5.000000
8	85	Art	4.0	25.5	Music	3.642857
9	95	Bob	3.0	63.5	Econ	9.071429
10	101	Joan	3.0	NaN	Math	NaN
11	107	Johannes	NaN	35.0	NaN	5.000000

[90]:
```
# and remember, Python and dataframes are actual functional in nature, so no
# no dataframes were harmed by the running of this Notebook's operations (:-))

sailors
```

[90]:

	sid	sname	rating	age	major
0	22	Dustin	7.0	45.0	CS
1	29	Brutus	1.0	33.0	EE
2	31	Lubber	8.0	55.5	Econ
3	32	Andy	8.0	25.5	Math
4	58	Rusty	10.0	35.0	CS
5	64	Horatio	7.0	35.0	CS
6	71	Zorba	10.0	16.0	CS
7	74	Horatio	9.0	35.0	Math
8	85	Art	4.0	25.5	Music
9	95	Bob	3.0	63.5	Econ
10	101	Joan	3.0	NaN	Math
11	107	Johannes	NaN	35.0	NaN

[ ]:
```

```