9 - Pandas-groupby

October 14, 2021

Table of Contents

[17]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib as mpl
import seaborn as sns
from numpy.random import randn

#import os
#for dirname, _, filenames in os.walk('/kaggle/input'):
#    for filename in filenames:
#        print(os.path.join(dirname, filename))

pd.set_option("display.precision", 1)
```

[18]:
```
df = sns.load_dataset("penguins")
```

[19]:
```
df.head()
```

[19]:

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	\
0	Adelie	Torgersen	39.1	18.7	181.0	
1	Adelie	Torgersen	39.5	17.4	186.0	
2	Adelie	Torgersen	40.3	18.0	195.0	
3	Adelie	Torgersen	NaN	NaN	NaN	
4	Adelie	Torgersen	36.7	19.3	193.0	

```
   body_mass_g     sex
0       3750.0    Male
1       3800.0  Female
2       3250.0  Female
3          NaN     NaN
4       3450.0  Female
```

1 Basics

[20]:
```
# A very basic example.  Split - apply - combine

# Split the data into groups
# Apply some function to each group
# Combine and return results

df.groupby('species').mean()
# no specific numeric column is specified the mean for all numeric columns is␣
 ↪calculated.
```

[20]:

species	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
Adelie	38.8	18.3	190.0	3700.7
Chinstrap	48.8	18.4	195.8	3733.1
Gentoo	47.5	15.0	217.2	5076.0

[21]:
```
# Specify the column(s) to group by and the column to ue for aggregation

df.groupby('species').body_mass_g.mean()
```

[21]:
```
species
Adelie       3700.7
Chinstrap    3733.1
Gentoo       5076.0
Name: body_mass_g, dtype: float64
```

[22]:
```
# To get the number of groups

df.groupby('species').ngroups

#x = df.groupby('species')
#len(x)
```

[22]:
```
3
```

[23]:
```
# How many observations in each group
x = df.groupby('species')
```

```
df.groupby('species').size()
```

[23]:
```
species
Adelie       152
Chinstrap     68
Gentoo       124
dtype: int64
```

[24]:
```
# Get groupby objects

x = df.groupby('species')
x.groups
```

[24]:
```
{'Adelie': [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, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58,
59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78,
79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98,
99, …], 'Chinstrap': [152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162,
163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178,
179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194,
195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210,
211, 212, 213, 214, 215, 216, 217, 218, 219], 'Gentoo': [220, 221, 222, 223,
224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239,
240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255,
256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271,
272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287,
288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303,
304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319,
…]}
```

[25]:
```
x.first()  # The equivalent of head() for a dataframe
```

[25]:

species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
Adelie	Torgersen	39.1	18.7	181.0	3750.0	Male
Chinstrap	Dream	46.5	17.9	192.0	3500.0	Female
Gentoo	Biscoe	46.1	13.2	211.0	4500.0	Female

[26]:
```
x.last()  # The equivalent of head() for a dataframe
```

[26]:
```
           island  bill_length_mm  bill_depth_mm  flipper_length_mm  \
species
Adelie      Dream            41.5           18.5              201.0
Chinstrap   Dream            50.2           18.7              198.0
Gentoo     Biscoe            49.9           16.1              213.0

           body_mass_g     sex
species
Adelie          4000.0    Male
Chinstrap       3775.0  Female
Gentoo          5400.0    Male
```

[27]:
```
# To retrieve one of the created groups

gentoo = x.get_group('Gentoo')
gentoo.head()
```

[27]:
```
    species  island  bill_length_mm  bill_depth_mm  flipper_length_mm  \
220  Gentoo  Biscoe            46.1           13.2              211.0
221  Gentoo  Biscoe            50.0           16.3              230.0
222  Gentoo  Biscoe            48.7           14.1              210.0
223  Gentoo  Biscoe            50.0           15.2              218.0
224  Gentoo  Biscoe            47.6           14.5              215.0

     body_mass_g     sex
220       4500.0  Female
221       5700.0    Male
222       4450.0  Female
223       5700.0    Male
224       5400.0    Male
```

[28]:
```
print(type(x))
print(type(gentoo))
```

```
<class 'pandas.core.groupby.generic.DataFrameGroupBy'>
<class 'pandas.core.frame.DataFrame'>
```

[29]:
```
# Display all methods
import IPython

methods = [method_name for method_name in dir(x)
           if callable(getattr(x, method_name)) & ~method_name.startswith('_')]

print(IPython.utils.text.columnize(methods))
```

```
agg         corrwith  diff        hist    ngroup      quantile  std
aggregate   count     ewm         idxmax  nth         rank      sum
all         cov       expanding   idxmin  nunique     resample  tail
any         cumcount  ffill       last    ohlc        rolling   take
```

```
apply       cummax     fillna     mad      pad          sample     transform
backfill    cummin     filter     max      pct_change   sem        tshift
bfill       cumprod    first      mean     pipe         shift      var
boxplot     cumsum     get_group  median   plot         size
corr        describe   head       min      prod         skew
```

2 Aggregation

[30]:
```
df.groupby('sex').body_mass_g.max()
```

[30]:
```
sex
Female    5200.0
Male      6300.0
Name: body_mass_g, dtype: float64
```

[31]:
```
df.groupby('sex').body_mass_g.min()
```

[31]:
```
sex
Female    2700.0
Male      3250.0
Name: body_mass_g, dtype: float64
```

[32]:
```
df.groupby('sex').body_mass_g.median()
```

[32]:
```
sex
Female    3650.0
Male      4300.0
Name: body_mass_g, dtype: float64
```

[33]:
```
df.groupby('sex').body_mass_g.count()
```

[33]:
```
sex
Female    165
Male      168
Name: body_mass_g, dtype: int64
```

2.1 With agg()

[34]:
```
# 1 - df.groupby('species').mean()              mean for all columns
# 2 - df.groupby('sex').body_mass_g.min()       mean for a specific column
# 3 -            NEXT                           multiple agregations for a␣
 ↪column(s)
```

[35]:
```
# there is a function called .agg() and it allows specifiying multiple␣
 ↪aggregation functions at once
```

```
df.groupby('sex').body_mass_g.agg(['max', 'min', 'count', 'median', 'mean'])
```

[35]:

sex	max	min	count	median	mean
Female	5200.0	2700.0	165	3650.0	3862.3
Male	6300.0	3250.0	168	4300.0	4545.7

[36]:
```
# with custom column name
df.groupby('sex').body_mass_g.agg(
    sex_max=('max'),
    sex_min=('min'),
)
```

[36]:

sex	sex_max	sex_min
Female	5200.0	2700.0
Male	6300.0	3250.0

[37]:
```
# Custom aggregation function
def categorize(x):
    m = x.mean()
    return True if m > 4000 else False

df.groupby('sex').body_mass_g.agg(['max', 'mean', categorize])
```

[37]:

sex	max	mean	categorize
Female	5200.0	3862.3	False
Male	6300.0	4545.7	True

[38]:
```
# Use lambda
df.groupby('sex').body_mass_g.agg(
    ['max', 'mean', lambda x: True if x.mean() > 4000 else False]
)
```

[38]:

sex	max	mean	<lambda_0>
Female	5200.0	3862.3	False
Male	6300.0	4545.7	True

[39]:
```
# REMINDER
# With a groupby a specific column for the aggregation does not have to be␣
 ↪specified.
# Without a column, it will perform the aggregation across all of the numeric␣
 ↪columns

df.groupby('sex').mean()
```

[39]:
```
        bill_length_mm  bill_depth_mm  flipper_length_mm  body_mass_g
sex
Female            42.1           16.4              197.4       3862.3
Male              45.9           17.9              204.5       4545.7
```

[40]:
```
df.groupby('sex').agg(['mean', 'median'])
```

[40]:
```
        bill_length_mm        bill_depth_mm        flipper_length_mm         \
                  mean median          mean median                mean median
sex
Female            42.1   42.8          16.4   17.0               197.4  193.0
Male              45.9   46.8          17.9   18.4               204.5  200.5

        body_mass_g
               mean  median
sex
Female      3862.3  3650.0
Male        4545.7  4300.0
```

3 Transforming Data

[41]:
```
# A lambda expression for Standardization.
standardization = lambda x: (x - x.mean()) / x.std()
```

[42]:
```
df.groupby('sex').body_mass_g.transform(standardization)
```

[42]:
```
0     -1.0e+00
1     -9.3e-02
2     -9.2e-01
4     -6.2e-01
5     -1.1e+00
        …
338    1.6e+00
340    1.5e+00
341    1.5e+00
342    2.0e+00
343    1.1e+00
Name: body_mass_g, Length: 333, dtype: float64
```

[43]:
```
df.groupby('sex').body_mass_g.apply(standardization)
```

[43]:
```
0     -1.0e+00
1     -9.3e-02
2     -9.2e-01
4     -6.2e-01
5     -1.1e+00
        …
```

```
338    1.6e+00
340    1.5e+00
341    1.5e+00
342    2.0e+00
343    1.1e+00
Name: body_mass_g, Length: 333, dtype: float64
```

4 Filtering data

[44]:
```
# How many rows fall into each island group?
df.groupby('island').size()
```

[44]:
```
island
Biscoe       168
Dream        124
Torgersen     52
dtype: int64
```

[45]:
```
# filter data to return all islands that have at least 100 observations.
df.groupby('island').filter(lambda x: len(x) >= 100)
```

[45]:
```
     species  island  bill_length_mm  bill_depth_mm  flipper_length_mm  \
20    Adelie  Biscoe            37.8           18.3              174.0
21    Adelie  Biscoe            37.7           18.7              180.0
22    Adelie  Biscoe            35.9           19.2              189.0
23    Adelie  Biscoe            38.2           18.1              185.0
24    Adelie  Biscoe            38.8           17.2              180.0
..       …       …               …              …                  …
339  Gentoo  Biscoe             NaN            NaN                NaN
340  Gentoo  Biscoe            46.8           14.3              215.0
341  Gentoo  Biscoe            50.4           15.7              222.0
342  Gentoo  Biscoe            45.2           14.8              212.0
343  Gentoo  Biscoe            49.9           16.1              213.0

     body_mass_g     sex
20        3400.0  Female
21        3600.0    Male
22        3800.0  Female
23        3950.0    Male
24        3800.0    Male
..           …     …
339          NaN     NaN
340       4850.0  Female
341       5750.0    Male
342       5200.0  Female
343       5400.0    Male
```

```
[292 rows x 7 columns]
```

5 Group by multiple categories

[46]:
```
# Creating a df that is a subset of penguins

small = df.loc[:, ['species', 'island', 'bill_depth_mm', 'bill_length_mm']]
small
```

[46]:
```
     species     island  bill_depth_mm  bill_length_mm
0     Adelie  Torgersen           18.7            39.1
1     Adelie  Torgersen           17.4            39.5
2     Adelie  Torgersen           18.0            40.3
3     Adelie  Torgersen            NaN             NaN
4     Adelie  Torgersen           19.3            36.7
..       …        …              …               …
339   Gentoo     Biscoe            NaN             NaN
340   Gentoo     Biscoe           14.3            46.8
341   Gentoo     Biscoe           15.7            50.4
342   Gentoo     Biscoe           14.8            45.2
343   Gentoo     Biscoe           16.1            49.9

[344 rows x 4 columns]
```

[47]:
```
# Grouping by multiple categories

small.groupby(['species', 'island']).mean()
```

[47]:
```
                    bill_depth_mm  bill_length_mm
species   island
Adelie    Biscoe             18.4            39.0
          Dream              18.3            38.5
          Torgersen          18.4            39.0
Chinstrap Dream              18.4            48.8
Gentoo    Biscoe             15.0            47.5
```

[48]:
```
df.groupby(['species', 'island']).mean()
```

[48]:
```
                    bill_length_mm  bill_depth_mm  flipper_length_mm  \
species   island
Adelie    Biscoe              39.0           18.4              188.8
          Dream               38.5           18.3              189.7
          Torgersen           39.0           18.4              191.2
Chinstrap Dream               48.8           18.4              195.8
Gentoo    Biscoe              47.5           15.0              217.2
```

```
                     body_mass_g
species   island
Adelie    Biscoe          3709.7
          Dream           3688.4
          Torgersen       3706.4
Chinstrap Dream           3733.1
Gentoo    Biscoe          5076.0
```

[49]:
```
# Group by multi column
df_groupby_multi = small.groupby(['species', 'island']).mean()
df_groupby_multi
```

[49]:
```
                     bill_depth_mm  bill_length_mm
species   island
Adelie    Biscoe              18.4            39.0
          Dream               18.3            38.5
          Torgersen           18.4            39.0
Chinstrap Dream               18.4            48.8
Gentoo    Biscoe              15.0            47.5
```

[50]:
```
df_groupby_multi.reset_index()
```

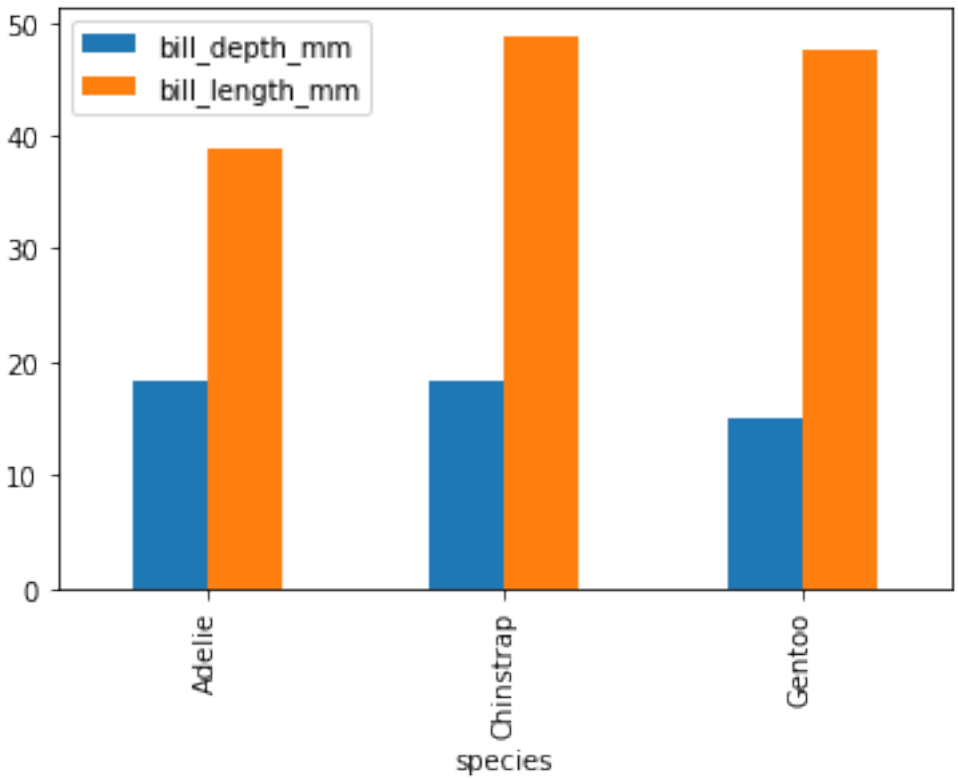
[50]:
```
     species     island  bill_depth_mm  bill_length_mm
0     Adelie     Biscoe           18.4            39.0
1     Adelie      Dream           18.3            38.5
2     Adelie  Torgersen           18.4            39.0
3  Chinstrap      Dream           18.4            48.8
4     Gentoo     Biscoe           15.0            47.5
```

[51]:
```
# A better way is to set as_index=False
small.groupby(['species', 'island'], as_index=False).mean()
```

[51]:
```
     species     island  bill_depth_mm  bill_length_mm
0     Adelie     Biscoe           18.4            39.0
1     Adelie      Dream           18.3            38.5
2     Adelie  Torgersen           18.4            39.0
3  Chinstrap      Dream           18.4            48.8
4     Gentoo     Biscoe           15.0            47.5
```

[52]:
```
small.groupby('species').mean().plot(kind='bar')  # This is actually a pandas␣
↪plot
```

[52]:
```
<AxesSubplot:xlabel='species'>
```



6 Group by numerical data using .cut() and .qcut()

[53]:
```
df['mass_group'] = pd.cut(df['body_mass_g'],
                          bins=[0, 3000, 4000, 5000, 10000],
                          labels=('small', 'medium','large', 'wow'))
df.head()
```

[53]:

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex	mass_group
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	Male	medium
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	Female	medium
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	Female	medium
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	Female	medium

[54]:
```
df.groupby('mass_group').agg(["mean", "median"])
```

[54]:

	bill_length_mm		bill_depth_mm		flipper_length_mm	body_mass_g	
	mean	median	mean	median	mean	median	mean
mass_group							
small	38.1	37.3	17.2	16.9	186.0	187.0	2900.0
medium	41.5	39.7	18.1	18.0	190.6	190.0	3576.1
large	45.0	45.3	16.6	15.2	206.1	209.0	4512.6
wow	49.3	49.3	15.6	15.7	221.1	221.0	5501.6

	body_mass_g
	median
mass_group	
small	2900.0
medium	3600.0
large	4500.0
wow	5500.0

[55]:
```
df.groupby(pd.cut(df['body_mass_g'],
                  bins=[0, 3000, 4000, 5000, 10000],
                  labels=('small', 'medium','large', 'wow'))).mean()
```

[55]:

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
body_mass_g				
small	38.1	17.2	186.0	2900.0
medium	41.5	18.1	190.6	3576.1
large	45.0	16.6	206.1	4512.6
wow	49.3	15.6	221.1	5501.6

[56]:
```
df.groupby(pd.qcut(df["body_mass_g"],4, duplicates="drop")).mean()
```

[56]:

	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
body_mass_g				
(2699.999, 3550.0]	39.9	17.7	188.6	3297.8
(3550.0, 4050.0]	43.2	18.5	192.7	3808.0
(4050.0, 4750.0]	44.4	16.8	203.9	4430.6
(4750.0, 6300.0]	48.5	15.5	219.3	5333.2

[57]:
```
# Just a note....
```

[58]:
```
df.groupby(['species', 'island']).bill_length_mm.sum().reset_index()
```

[58]:

	species	island	bill_length_mm
0	Adelie	Biscoe	1714.9
1	Adelie	Dream	2156.1
2	Adelie	Torgersen	1986.5
3	Chinstrap	Dream	3320.7
4	Gentoo	Biscoe	5843.1

[59]:
```
# A different way to write he same code
df.groupby(['species', 'island'])['bill_length_mm'].sum().reset_index()
```

[59]:

	species	island	bill_length_mm
0	Adelie	Biscoe	1714.9
1	Adelie	Dream	2156.1
2	Adelie	Torgersen	1986.5
3	Chinstrap	Dream	3320.7
4	Gentoo	Biscoe	5843.1

7 Return to the Beer notebook and complete part 2