1 Summarising data by groups in Pandas using pivot tables and groupby

Pandas offers two methods of summarising data - groupby and pivot_table*. The data produced can be the same but the format of the output may differ.

*pivot_table summarises data. There is a similar command, pivot, which we will use in the next section which is for reshaping data.

As usual let's start by creating a dataframe.

```
import pandas as pd
df = pd.DataFrame()

names = ['Gandolf',
         'Gimli',
         'Frodo',
         'Legolas',
         'Bilbo',
         'Sam',
         'Pippin',
         'Boromir',
         'Aragorn',
         'Galadriel',
         'Meriadoc']
types = ['Wizard',
         'Dwarf',
         'Hobbit',
         'Elf',
         'Hobbit',
         'Hobbit',
         'Hobbit',
         'Man',
         'Man',
         'Elf',
         'Hobbit']
magic = [10, 1, 4, 6, 4, 2, 0, 0, 2, 9, 0]
aggression = [7, 10, 2, 5, 1, 6, 3, 8, 7, 2, 4]
stealth = [8, 2, 5, 10, 5, 4 ,5, 3, 9, 10, 6]


df['names'] = names
df['type'] = types
df['magic_power'] = magic
df['aggression'] = aggression
df['stealth'] = stealth
```

1.1 Pivot tables

To return the median values by type:

```
import numpy as np # we will use a numpy method to summarise data

pivot = df.pivot_table(index=['type'],
                       values=['magic_power','aggression', 'stealth'],
                       aggfunc=[np.mean],
                       margins=True) # margins summarises all

# note: addgunc can be any valid function that can act on data provided
```

```
print (pivot)

OUT:

             mean
       aggression magic_power    stealth
type
Dwarf        10.0    1.000000   2.000000
Elf           3.5    7.500000  10.000000
Hobbit        3.2    2.000000   5.000000
Man           7.5    1.000000   6.000000
Wizard        7.0   10.000000   8.000000
All           5.0    3.454545   6.090909
```

Or we may group by more than one index. In this case we'll return the average and summed values by type and magical power:

```
pivot = df.pivot_table(index=['type','magic_power'],
                       values=['aggression', 'stealth'],
                       aggfunc=[np.mean,np.sum],
                       margins=True) # margins summarises all

print (pivot)

OUT:

                        mean                   sum
                  aggression    stealth aggression stealth
type   magic_power
Dwarf  1                10.0   2.000000         10       2
Elf    6                 5.0  10.000000          5      10
       9                 2.0  10.000000          2      10
Hobbit 0                 3.5   5.500000          7      11
       2                 6.0   4.000000          6       4
       4                 1.5   5.000000          3      10
Man    0                 8.0   3.000000          8       3
       2                 7.0   9.000000          7       9
Wizard 10                7.0   8.000000          7       8
All                      5.0   6.090909         55      67
```

1.2 Groupby

Grouby is a very powerful method in Pandas which we shall return to in the next section. Here we will use groupby simply to summarise data.

```
print(df.groupby('type').median())

OUT:

        magic_power  aggression  stealth
type
Dwarf           1.0        10.0      2.0
Elf             7.5         3.5     10.0
Hobbit          2.0         3.0      5.0
Man             1.0         7.5      6.0
Wizard         10.0         7.0      8.0
```

Instead of built in methods we can also apply user-defined functions. To illustrate we'll define a simple

function to return the lower quartile.

```
def my_func(x):
    return (x.quantile(0.25))

print(df.groupby('type').apply(my_func))

# Note we need not apply a lambda function
# We may apply any user-defined function
```

OUT:

```
0.25    magic_power  aggression  stealth
type
Dwarf          1.00       10.00      2.0
Elf            6.75        2.75     10.0
Hobbit         0.00        2.00      5.0
Man            0.50        7.25      4.5
Wizard        10.00        7.00      8.0
```

As with pivot-table we can have more than one index column.

```
print(df.groupby(['type','magic_power']).median())
```

OUT:

```
                    aggression  stealth
type   magic_power
Dwarf  1                  10.0      2.0
Elf    6                   5.0     10.0
       9                   2.0     10.0
Hobbit 0                   3.5      5.5
       2                   6.0      4.0
       4                   1.5      5.0
Man    0                   8.0      3.0
       2                   7.0      9.0
Wizard 10                  7.0      8.0
```

Or we can return just selected data columns.

```
print(df.groupby('type').median()[['magic_power','stealth']])
```

OUT:

```
        magic_power  stealth
type
Dwarf           1.0      2.0
Elf             7.5     10.0
Hobbit          2.0      5.0
Man             1.0      6.0
Wizard         10.0      8.0
```

To return multiple types of results we use the *agg* argument.

```
print(df.groupby('type').agg([min, max]))
```

OUT:

```
            names          magic_power     aggression     stealth
              min      max         min max        min max     min max
type
Dwarf       Gimli    Gimli           1   1         10  10       2   2
```

```
Elf     Galadriel  Legolas          6   9          2   5      10  10
Hobbit      Bilbo      Sam          0   4          1   6       4   6
Man       Aragorn  Boromir          0   2          7   8       3   9
Wizard    Gandolf  Gandolf         10  10          7   7       8   8
```

1.2.1 Pandas built-in groupby functions

Remember that apply can be used to apply any user-defined function

.all # Boolean True if all true

.any # Boolean True if any true

.count count of non null values

.size size of group including null values

.max

.min

.mean

.median

.sem

.std

.var

.sum

.prod

.quantile

.agg(functions) # for multiple outputs

.apply(func)

.last # last value

.nth # nth row of group