

Python - Analiza danych z modulem PANDAS

www.udemy.com (<http://www.udemy.com>) (R)

LAB - S06-L007 - przetwarzanie obiektu grupy

1. Zaimportuj moduł pandas i numpy nadaj im standardowe aliasy. Zaimportuj też datetime, timedelta i time, możesz skorzystać z poniższych poleceń:

```
from datetime import datetime
from datetime import timedelta
import time
```

2. Do wykonania zadań będziemy korzystać z danych dotyczących maratonów. Uruchom poniższy kod, który przygotuje zmienną df o odpowiedniej strukturze:

```
df = pd.read_csv('./marathon_results_2016.csv', index_col='Bib',
                usecols=['Bib', '40K', 'Half', 'Pace', 'Age', 'M/F', 'Country', 'State', 'City'
                ])

df['40K'] = df['40K'].apply(pd.to_timedelta)
df['Half'] = df['Half'].apply(pd.to_timedelta)

df['TotalSeconds'] = df['40K'].apply(lambda x: timedelta.total_seconds(
x))
df['HalfSeconds'] = df['Half'].apply(lambda x: timedelta.total_seconds(
x))

df.head()
```

3. W zmiennej **group_city** zapisz wynik grupowania data frame **df** ze względu na kolumnę **City**

Uwaga! W rozwiązaniu poniżej ograniczam ilość pobranych wierszy do 10. Wszystko po to, aby wyniki zajmowały rozsądną ilość stron w tym PDF;)

4. Napisz pętlę, która wyświetli nazwę miasta z każdego wiersza grupy **group_city**
5. Napisz pętlę, która wyświetli nazwę miasta z każdego wiersza grupy **group_city** i obok ilość uczestników maratonu pochodzących z tego miasta
6. Jeśli jeszcze tego nie zrobiłeś wykonaj poprzednie polecenie korzystając z dwóch zmiennych iterujących po obiekcie **group_city**
7. Nadal korzystając z pętli wyświetl z obiektu grupy **group_city** nazwę miasta i dane pochodzące z tego wiersza należącego do grupy, w którym czas **TotalSeconds** był najkrótszy.
8. Zadeklaruj zmienną **the_best_per_city** jako pusty obiekt **DataFrame**. Korzystając z pętli for iterującej po obiekcie grupy **group_city** skopiuj do **the_best_per_city** wiersze maratończyków, którzy osiągnęli najkrótszy czas biegu (kolumna **TotalSeconds**). Następnie wyświetl **the_best_per_city**

Dane pochodzą z <https://github.com/llimllib/bostonmarathon> (<https://github.com/llimllib/bostonmarathon>)
<https://www.kaggle.com/rojour/boston-marathon-2016-finishers-analysis/data> (<https://www.kaggle.com/rojour/boston-marathon-2016-finishers-analysis/data>)

Rozwiązania:

Poniżej znajdują się propozycje rozwiązań zadań. Prawdopodobnie istnieje wiele dobrych rozwiązań, dlatego jeżeli rozwiązujesz zadania samodzielnie, to najprawdopodobniej zrobisz to inaczej, może nawet lepiej :) Możesz pochwalić się swoimi rozwiązaniami w sekcji Q&A

```
In [35]: import pandas as pd
import numpy as np
from datetime import datetime
from datetime import timedelta
import time
```

```
In [36]: df = pd.read_csv('./marathon_results_2016.csv', index_col='Bib',
                        usecols=['Bib', '40K', 'Half', 'Pace', 'Age', 'M/F', 'Country', 'State', 'C

df['40K'] = df['40K'].apply(pd.to_timedelta)
df['Half'] = df['Half'].apply(pd.to_timedelta)

df['TotalSeconds'] = df['40K'].apply(lambda x: timedelta.total_seconds(x))
df['HalfSeconds'] = df['Half'].apply(lambda x: timedelta.total_seconds(x))

df.head()
```

```
Out[36]:
```

	Age	M/F	City	State	Country	Half	40K	Pace	TotalSeconds	HalfSeconds
Bib										
5	21	M	Addis Ababa	NaN	ETH	01:06:45	02:05:59	0:05:04	7559.0	4005.0
1	26	M	Ambo	NaN	ETH	01:06:46	02:05:59	0:05:06	7559.0	4006.0
6	31	M	Addis Ababa	NaN	ETH	01:06:44	02:06:47	0:05:07	7607.0	4004.0
11	33	M	Kitale	NaN	KEN	01:06:46	02:06:47	0:05:07	7607.0	4006.0
14	23	M	Eldoret	NaN	KEN	01:06:46	02:08:11	0:05:11	7691.0	4006.0

```
In [37]: df = df.head(10)
group_city = df.groupby(by="City")
```

```
In [38]: for city in group_city:
print(city[0])
```

```
Addis Ababa
Ambo
Dallas
Eldoret
Kitale
Nijmegen
```

```
In [39]: for city in group_city:
print(city[0], len(city[1]))
```

```
Addis Ababa 3
Ambo 1
Dallas 1
Eldoret 3
Kitale 1
Nijmegen 1
```

```
In [40]: for city, city_data in group_city:
        print(city, len(city_data))
```

```
Addis Ababa 3
Ambo 1
Dallas 1
Eldoret 3
Kitale 1
Nijmegen 1
```

```
In [41]: for city, city_data in group_city:
        print(city, city_data.nsmallest(1, "TotalSeconds"))
```

```
Addis Ababa      Age M/F      City State Country      Half      40K      Pace
\
Bib
5      21      M      Addis Ababa      NaN      ETH 01:06:45 02:05:59 0:05:04

      TotalSeconds      HalfSeconds
Bib
5      7559.0      4005.0
Ambo      Age M/F      City State Country      Half      40K      Pace      TotalSeconds
\
Bib
1      26      M      Ambo      NaN      ETH 01:06:46 02:05:59 0:05:06      7559.0

      HalfSeconds
Bib
1      4006.0
Dallas      Age M/F      City State Country      Half      40K      Pace      TotalSeconds
\
Bib
43      28      M      Dallas      TX      USA 01:07:57 02:12:47 0:05:25      7967.0

      HalfSeconds
Bib
43      4077.0
Eldoret      Age M/F      City State Country      Half      40K      Pace      TotalSeconds
\
Bib
14      23      M      Eldoret      NaN      KEN 01:06:46 02:08:11 0:05:11      7691.0

      HalfSeconds
Bib
14      4006.0
Kitale      Age M/F      City State Country      Half      40K      Pace      TotalSeconds
\
Bib
11      33      M      Kitale      NaN      KEN 01:06:46 02:06:47 0:05:07      7607.0

      HalfSeconds
Bib
11      4006.0
Nijmegen      Age M/F      City State Country      Half      40K      Pace      TotalSeconds
\
Bib
17      27      M      Nijmegen      NaN      NED 01:06:47 02:08:38 0:05:16      7718.0

      HalfSeconds
Bib
17      4007.0
```

```
In [42]: the_best_per_city = pd.DataFrame()

for city, city_data in group_city:
    the_best_per_city = the_best_per_city.append(city_data.nsmallest(1, "TotalSeconds"))

the_best_per_city.head(5)
```

Out[42]:

	Age	M/F	City	State	Country	Half	40K	Pace	TotalSeconds	HalfSeconds
Bib										
5	21	M	Addis Ababa	NaN	ETH	01:06:45	02:05:59	0:05:04	7559.0	4005.0
1	26	M	Ambo	NaN	ETH	01:06:46	02:05:59	0:05:06	7559.0	4006.0
43	28	M	Dallas	TX	USA	01:07:57	02:12:47	0:05:25	7967.0	4077.0
14	23	M	Eldoret	NaN	KEN	01:06:46	02:08:11	0:05:11	7691.0	4006.0
11	33	M	Kitale	NaN	KEN	01:06:46	02:06:47	0:05:07	7607.0	4006.0