Stanford Open Policing Project dataset

ANALYZING POLICE ACTIVITY WITH PANDAS

Kevin Markham
Founder, Data School

Introduction to the dataset

- Traffic stops by police officers

- Download data for any state:

https://openpolicing.stanford.edu/

Preparing the data

- Examine the data

- Clean the data

```python
import pandas as pd
ri = pd.read_csv('police.csv')
ri.head(3)
```

```
   state   stop_date stop_time   county_name driver_gender driver_race
0     RI  2005-01-04     12:55           NaN             M        White
1     RI  2005-01-23     23:15           NaN             M        White
2     RI  2005-02-17     04:15           NaN             M        White
```

- Each row represents one traffic stop

- `NaN` indicates a missing value

Locating missing values (1)

```
ri.isnull()
```

```
   state stop_date stop_time county_name driver_gender
0  False     False     False        True         False
1  False     False     False        True         False
2  False     False     False        True         False
...
```

Locating missing values (2)

```
ri.isnull().sum()
```

```
state                 0
stop_date             0
stop_time             0
county_name       91741
driver_gender      5205
...
```

- `.sum()` calculates the sum of each column

- `True = 1`, `False = 0`

Dropping a column

```
ri.isnull().sum()
```

```
state               0
stop_date           0
stop_time           0
county_name     91741
driver_gender    5205
driver_race      5202
...
```

```
ri.shape
```

```
(91741, 15)
```

- `county_name` column only contains missing values

- Drop `county_name` using the `.drop()` method

```
ri.drop('county_name',
    axis='columns', inplace=True)
```

Dropping rows

- `.dropna()` : Drop rows based on the presence of missing values

```
ri.head()
```

```
   state    stop_date stop_time driver_gender driver_race
0     RI   2005-01-04     12:55             M       White
1     RI   2005-01-23     23:15             M       White
2     RI   2005-02-17     04:15             M       White
3     RI   2005-02-20     17:15             M       White
4     RI   2005-02-24     01:20             F       White
```

```
ri.dropna(subset=['stop_date', 'stop_time'], inplace=True)
```

Let's practice!

datacamp

Using proper data types

ANALYZING POLICE ACTIVITY WITH PANDAS

Kevin Markham
Founder, Data School

Examining the data types

```
ri.dtypes
```

```
stop_date            object
stop_time            object
driver_gender        object
...                     ...
stop_duration        object
drugs_related_stop     bool
district             object
```

- `object` : Python strings (or other Python objects)

- `bool` : `True` and `False` values

- Other types: `int` , `float` , `datetime` , `category`

Why do data types matter?

- Affects which operations you can perform

- Avoid storing data as strings (when possible)
  - `int` , `float` : enables mathematical operations

  - `datetime` : enables date-based attributes and methods

  - `category` : uses <mark>less memory and runs faster</mark>

  - `bool` : enables logical and mathematical operations

Fixing a data type

```
apple
```

```
        date     time    price
0   2/13/18    16:00   164.34
1   2/14/18    16:00   167.37
2   2/15/18    16:00   172.99
```

```
apple.price.dtype
```

```
dtype('O')
```

```
apple['price'] =
    apple.price.astype('float')
```

```
apple.price.dtype
```

```
dtype('float64')
```

- Dot notation: `apple.price`

- Bracket notation:
  `apple['price']`

Let's practice!

ANALYZING POLICE ACTIVITY WITH PANDAS

Creating a DatetimeIndex

ANALYZING POLICE ACTIVITY WITH PANDAS

Kevin Markham
Founder, Data School

Using datetime format

```
ri.head(3)
```

```
     stop_date stop_time driver_gender driver_race
0   2005-01-04     12:55             M        White
1   2005-01-23     23:15             M        White
2   2005-02-17     04:15             M        White
```

```
ri.dtypes
```

```
stop_date            object
stop_time            object
driver_gender        object
driver_race          object
...
```

1. Combine `stop_date` and `stop_time` into one column

2. Convert it to `datetime` format

Combining object columns

apple

```
      date    time    price
0  2/13/18   16:00   164.34
1  2/14/18   16:00   167.37
2  2/15/18   16:00   172.99
```

apple.date.str.replace('/', '-')

```
0     2-13-18
1     2-14-18
2     2-15-18
Name: date, dtype: object
```

combined =
    apple.date.str.cat(apple.time, sep=' ')

combined

```
0     2/13/18 16:00
1     2/14/18 16:00
2     2/15/18 16:00
Name: date, dtype: object
```

Converting to datetime format

```
apple['date_and_time'] = pd.to_datetime(combined)
apple
```

```
       date    time    price        date_and_time
0   2/13/18   16:00   164.34  2018-02-13 16:00:00
1   2/14/18   16:00   167.37  2018-02-14 16:00:00
2   2/15/18   16:00   172.99  2018-02-15 16:00:00
```

```
apple.dtypes
```

```
date                       object
time                       object
price                      float64
date_and_time      datetime64[ns]
```

Setting the index

```
apple.set_index('date_and_time', inplace=True)
apple
```

```
                      date    time   price
date_and_time
2018-02-13 16:00:00  2/13/18  16:00  164.34
2018-02-14 16:00:00  2/14/18  16:00  167.37
2018-02-15 16:00:00  2/15/18  16:00  172.99
```

```
apple.index
```

```
DatetimeIndex(['2018-02-13 16:00:00', '2018-02-14 16:00:00',
               '2018-02-15 16:00:00'],
              dtype='datetime64[ns]', name='date_and_time', freq=None)
```

Let's practice!

ANALYZING POLICE ACTIVITY WITH PANDAS