ANALYZING POLICE ACTIVITY WITH PANDAS

Does time of day affect arrest rate?

Kevin Markham
Founder, Data School

Accessing datetime attributes (1)

```
apple

    price    volume       date_and_time
0  174.35  20567800 2018-01-08 16:00:00
1  174.33  21584000 2018-01-09 16:00:00
2  155.15  54390500 2018-02-08 16:00:00
3  156.41  70672600 2018-02-09 16:00:00
4  176.94  23774100 2018-03-08 16:00:00
5  179.98  32185200 2018-03-09 16:00:00

apple.dtypes

price                   float64
volume                    int64
date_and_time    datetime64[ns]
dtype: object

apple.date_and_time.dt.month

0    1
1    1
2    2
3    2
4    3
5    3
```

Accessing datetime attributes (2)

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

apple.index

DatetimeIndex(['2018-01-08 16:00:00', '2018-01-09 16:00:00',
               '2018-02-08 16:00:00', '2018-02-09 16:00:00',
               '2018-03-08 16:00:00', '2018-03-09 16:00:00'],
              dtype='datetime64[ns]', name='date_and_time', freq=None)

apple.index.month
Int64Index([1, 1, 2, 2, 3, 3], dtype='int64', name='date_and_time')
```

- dt accessor is not used with a DatetimeIndex

Calculating the monthly mean price

```
apple.price.mean()
169.52666666666667
```

- apple.groupby('month').price.mean() is invalid

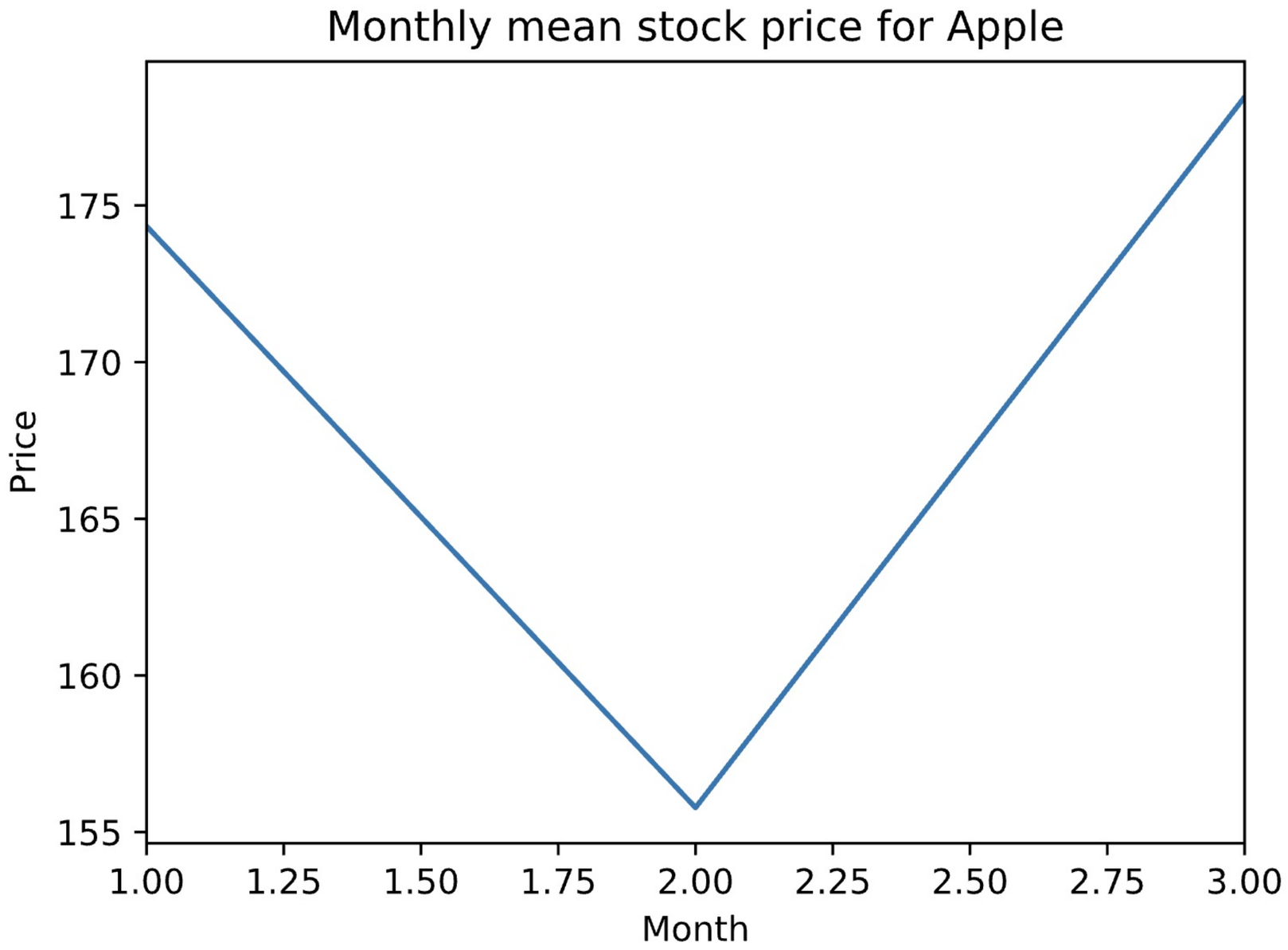
```
apple.groupby(apple.index.month).price.mean()

date_and_time
1    174.34
2    155.78
3    178.46
Name: price, dtype: float64

monthly_price = apple.groupby(apple.index.month).price.mean()
```

Plotting the monthly mean price

```
import matplotlib.pyplot as plt

monthly_price.plot()
```

- Line plot: Series index on x-axis, Series values on y-axis

```
plt.xlabel('Month')
plt.ylabel('Price')
plt.title('Monthly mean stock price for Apple')

plt.show()
```


Monthly mean stock price for Apple
Price
175
170
165
160
155
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Month

ANALYZING POLICE ACTIVITY WITH PANDAS

Let's practice!

ANALYZING POLICE ACTIVITY WITH PANDAS

Are drug-related stops on the rise?

Resampling the price

```
apple
                     price    volume
date_and_time
2018-01-08 16:00:00  174.35  20567800
2018-01-09 16:00:00  174.33  21584000
2018-02-08 16:00:00  155.15  54390500
2018-02-09 16:00:00  156.41  70672600
2018-03-08 16:00:00  176.94  23774100
2018-03-09 16:00:00  179.98  32185200

apple.groupby(apple.index.month).price.mean()

date_and_time
1    174.34
2    155.78
3    178.46
Name: price, dtype: float64

apple.price.resample('M').mean()

date_and_time
2018-01-31    174.34
2018-02-28    155.78
2018-03-31    178.46
Freq: M, Name: price, dtype: float64
```

Resampling the volume

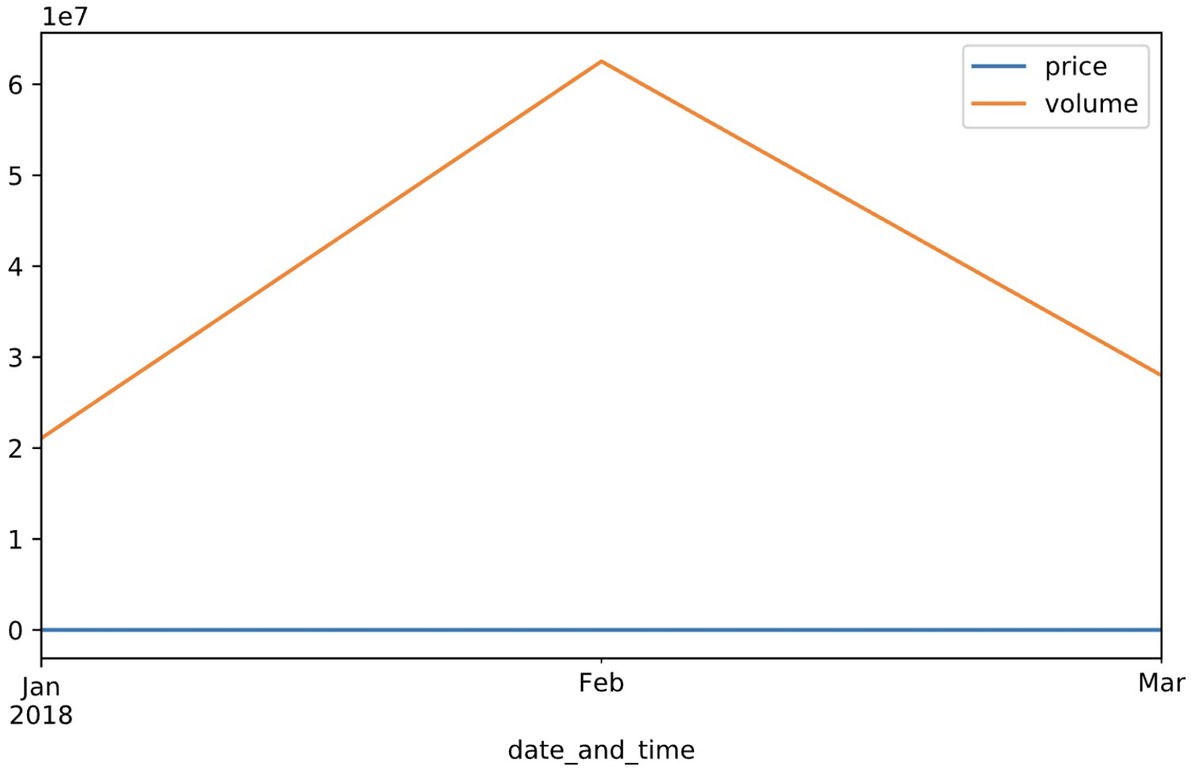
```
apple
                     price    volume
date_and_time
2018-01-08 16:00:00  174.35  20567800
2018-01-09 16:00:00  174.33  21584000
2018-02-08 16:00:00  155.15  54390500
2018-02-09 16:00:00  156.41  70672600
2018-03-08 16:00:00  176.94  23774100
2018-03-09 16:00:00  179.98  32185200

apple.volume.resample('M').mean()

date_and_time
2018-01-31    21075900
2018-02-28    62531550
2018-03-31    27979650
Freq: M, Name: volume, dtype: int64
```

Concatenating price and volume

```
monthly_price = apple.price.resample('M').mean()

monthly_volume = apple.volume.resample('M').mean()

pd.concat([monthly_price, monthly_volume], axis='columns')

                price    volume
date_and_time
2018-01-31     174.34  21075900
2018-02-28     155.78  62531550
2018-03-31     178.46  27979650

monthly = pd.concat([monthly_price, monthly_volume], axis='columns')
```

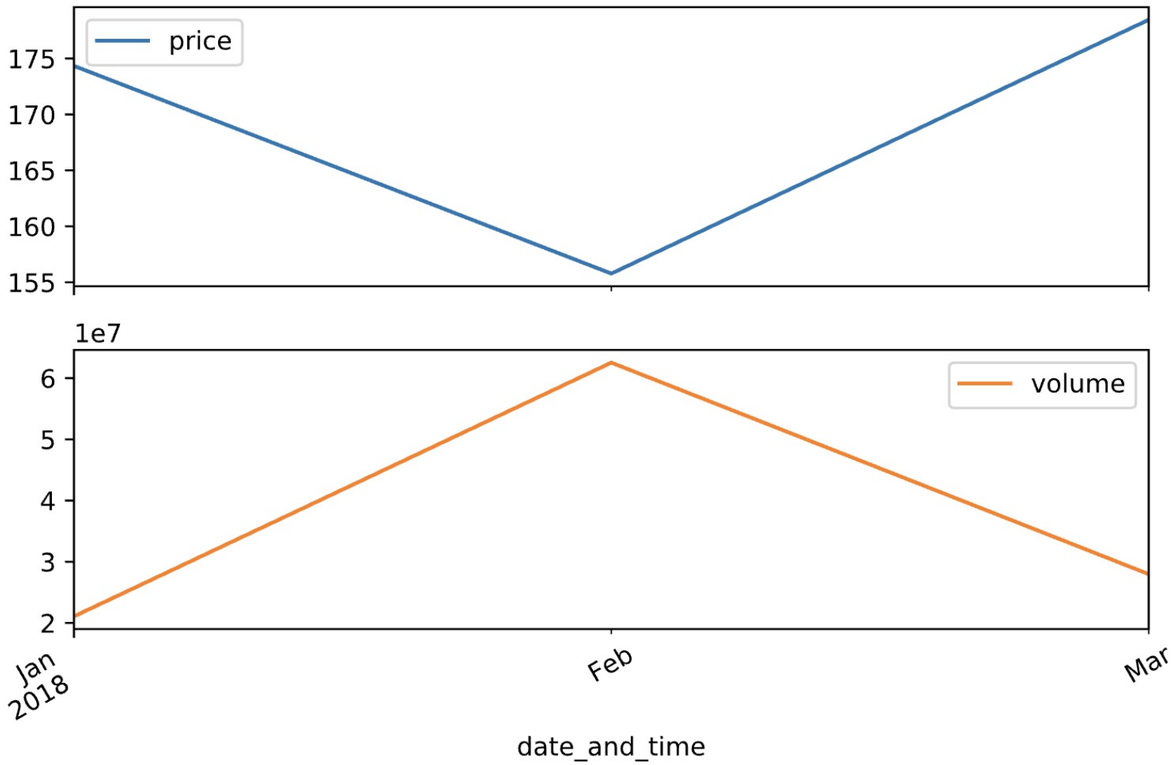
Plotting price and volume (1)

```
monthly.plot()
plt.show()
```



Plotting price and volume (2)

```
monthly.plot(subplots=True)
plt.show()
```



ANALYZING POLICE ACTIVITY WITH PANDAS

Let's practice!

ANALYZING POLICE ACTIVITY WITH PANDAS

What violations are caught in each district?

Computing a frequency table

```
pd.crosstab(ri.driver_race, ri.driver_gender)

driver_gender      F      M
driver_race
Asian            551   1838
Black           2681   9604
Hispanic        1953   7774
Other             53    212
White          18536  43334
```

- Frequency table: Tally of how many times each combination of values occurs

```
ri[(ri.driver_race == 'Asian') & (ri.driver_gender == 'F')].shape
(551, 14)
```

- driver_race is along the index, driver_gender is along the columns

```
table = pd.crosstab(ri.driver_race, ri.driver_gender)
```

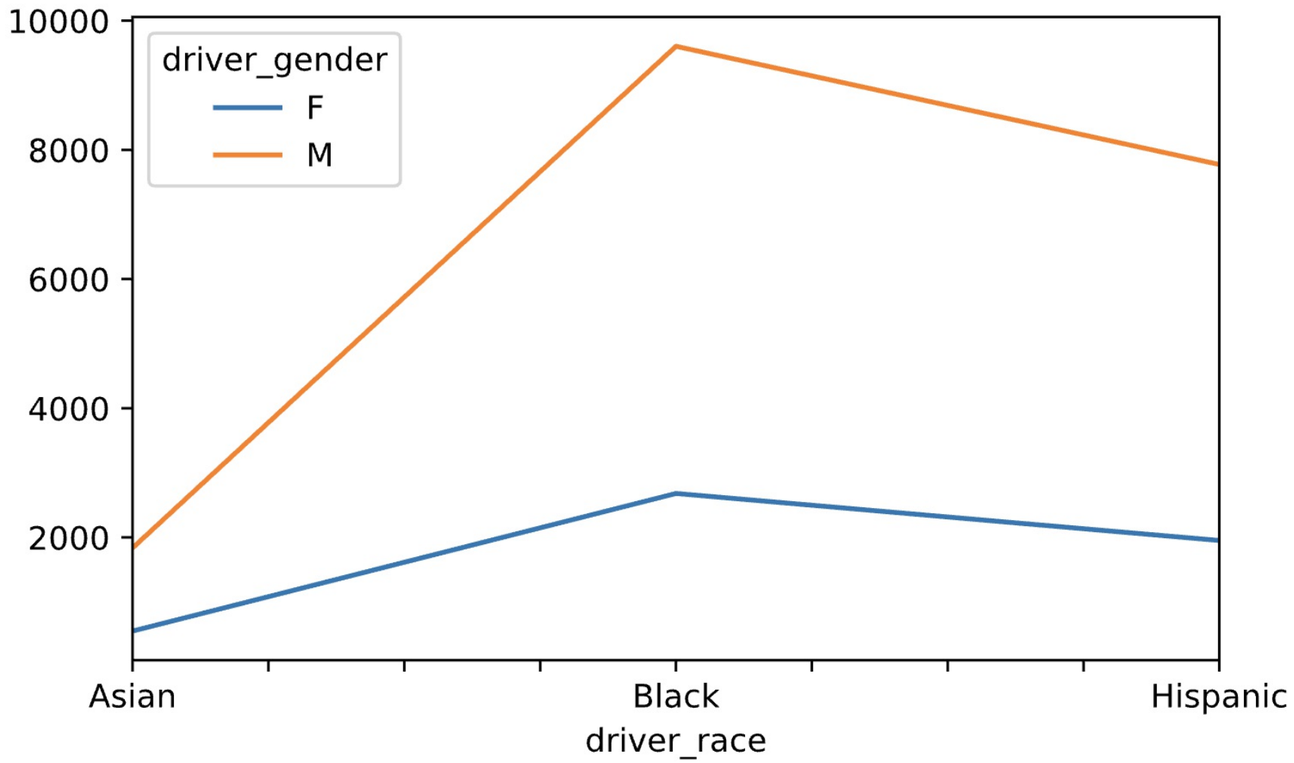
Selecting a DataFrame slice

- loc accessor: Select from a DataFrame by label

```
table

driver_gender      F      M
driver_race
Asian            551   1838
Black           2681   9604
Hispanic        1953   7774
Other             53    212
White          18536  43334

table.loc['Asian':'Hispanic']

driver_gender     F     M
driver_race
Asian           551  1838
Black          2681  9604
Hispanic       1953  7774

table = table.loc['Asian':'Hispanic']
```

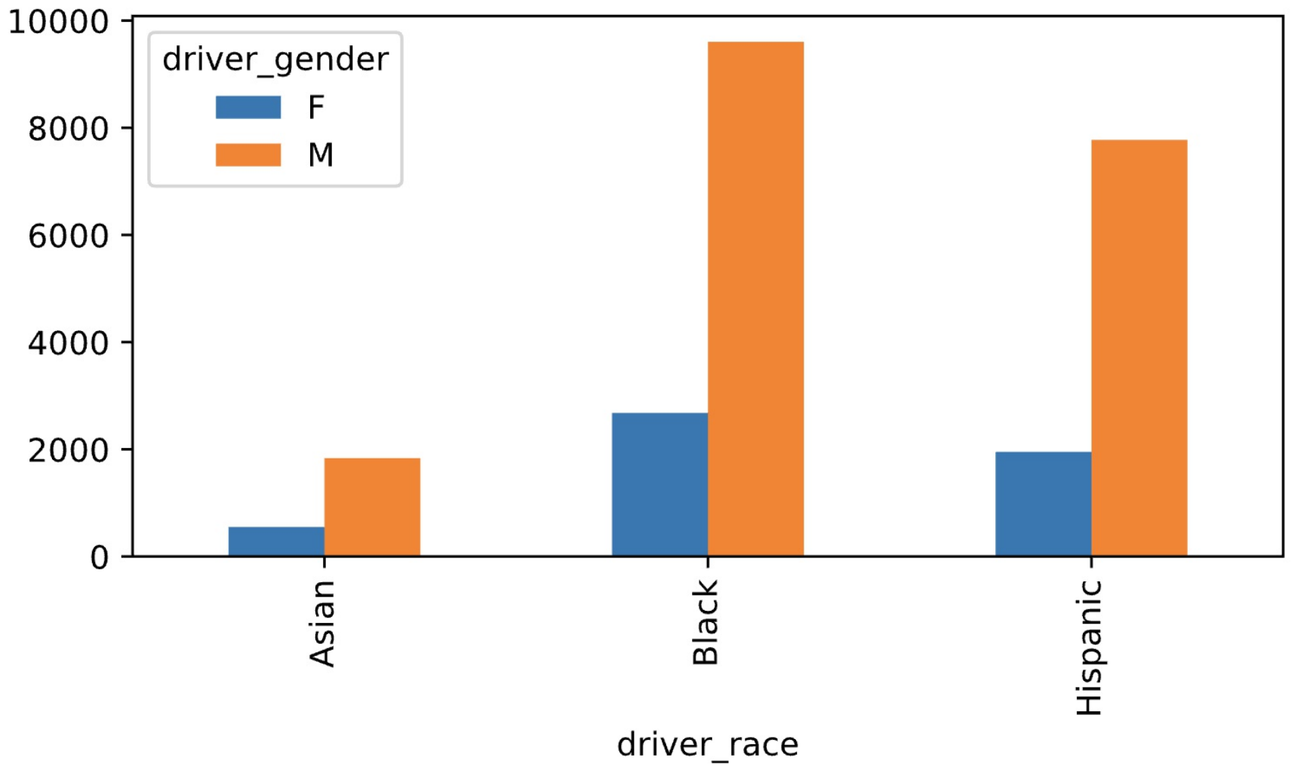
Creating a line plot

```
table.plot()
plt.show()
```



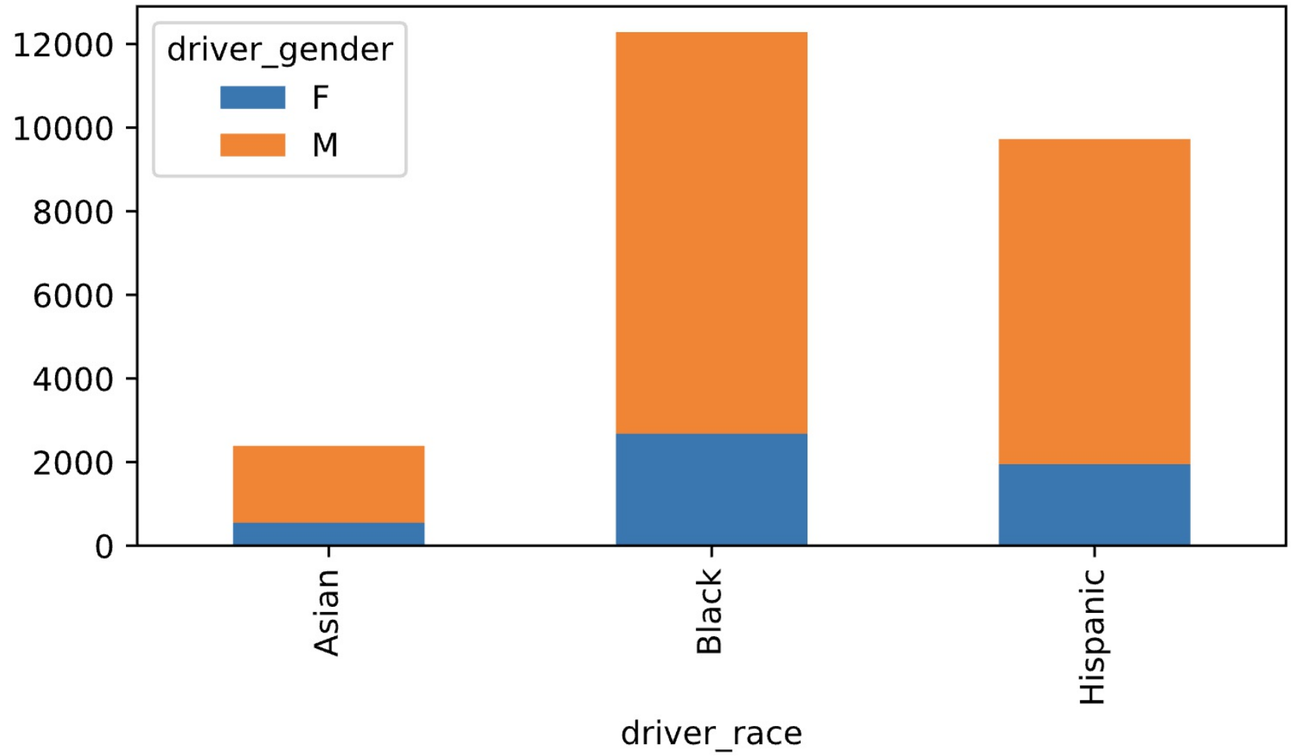
Creating a bar plot

```
table.plot(kind='bar')
plt.show()
```



Stacking the bars

```
table.plot(kind='bar', stacked=True)
plt.show()
```



ANALYZING POLICE ACTIVITY WITH PANDAS

Let's practice!

ANALYZING POLICE ACTIVITY WITH PANDAS

How long might you be stopped for a violation?

Analyzing an object column

```
apple

                     price    volume change
date_and_time
2018-01-08 16:00:00  174.35  20567800   down
2018-01-09 16:00:00  174.33  21584000   down
2018-02-08 16:00:00  155.15  54390500   down
2018-02-09 16:00:00  156.41  70672600     up
2018-03-08 16:00:00  176.94  23774100     up
2018-03-09 16:00:00  179.98  32185200     up
```

- Create a Boolean column: `True` if the price went up, and `False` otherwise
- Calculate how often the price went up by taking the column mean

```
apple.change.dtype
dtype('O')
```

- `astype()` can't be used in this case

Mapping one set of values to another

- Dictionary maps the values you have to the values you want

```
mapping = {'up':True, 'down':False}

apple['is_up'] = apple.change.map(mapping)

apple

                     price    volume change  is_up
date_and_time
2018-01-08 16:00:00  174.35  20567800   down  False
2018-01-09 16:00:00  174.33  21584000   down  False
2018-02-08 16:00:00  155.15  54390500   down  False
2018-02-09 16:00:00  156.41  70672600     up   True
2018-03-08 16:00:00  176.94  23774100     up   True
2018-03-09 16:00:00  179.98  32185200     up   True

apple.is_up.mean()
0.5
```

Calculating the search rate

- Visualize how often searches were performed after each type of violation

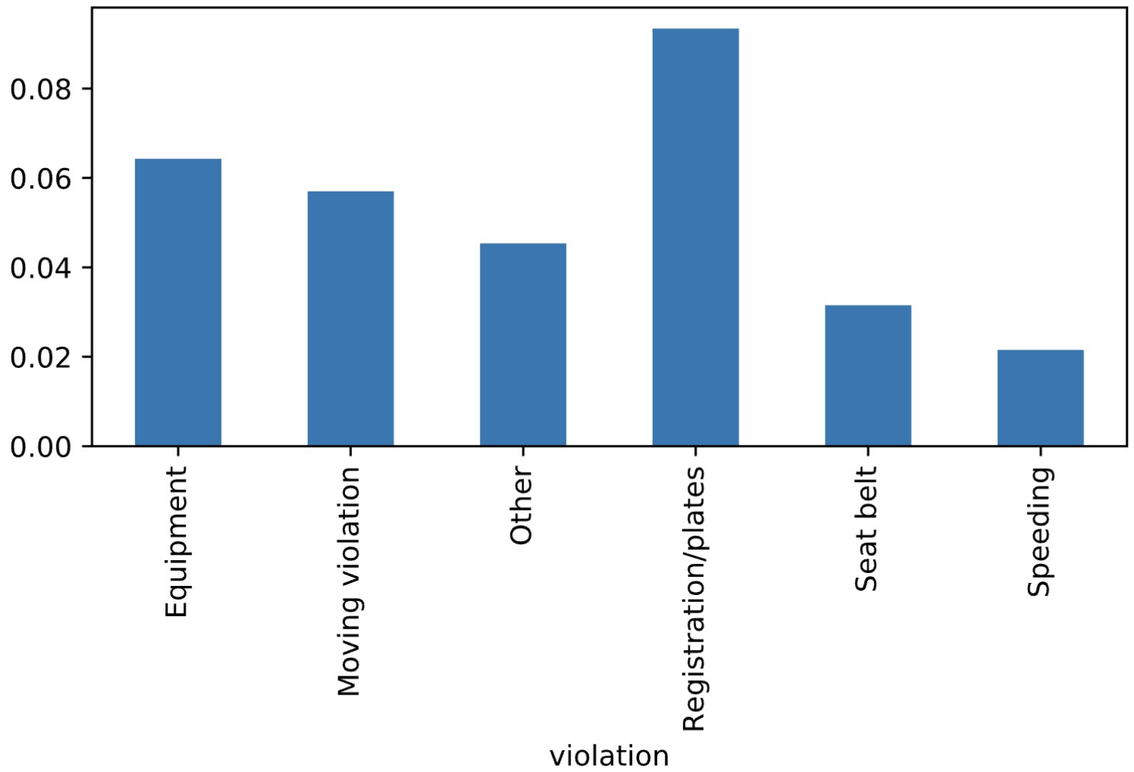
```
ri.groupby('violation').search_conducted.mean()

violation
Equipment              0.064280
Moving violation       0.057014
Other                  0.045362
Registration/plates    0.093438
Seat belt              0.031513
Speeding               0.021560
Name: search_conducted, dtype: float64
```

- Returns a Series sorted in alphabetical order

```
search_rate = ri.groupby('violation').search_conducted.mean()
```

Creating a bar plot

```
search_rate.plot(kind='bar')
plt.show()
```



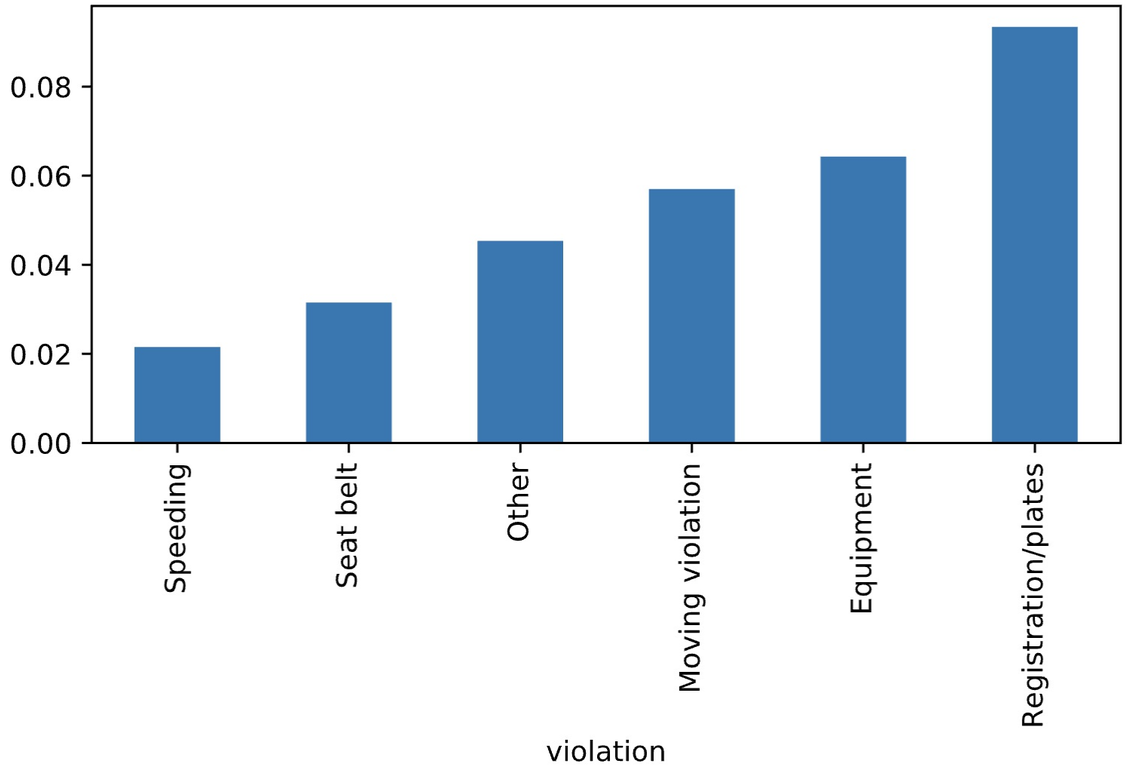
Ordering the bars (1)

- Order the bars from left to right by size

```
search_rate.sort_values()

violation
Speeding               0.021560
Seat belt              0.031513
Other                  0.045362
Moving violation       0.057014
Equipment              0.064280
Registration/plates    0.093438
Name: search_conducted, dtype: float64
```

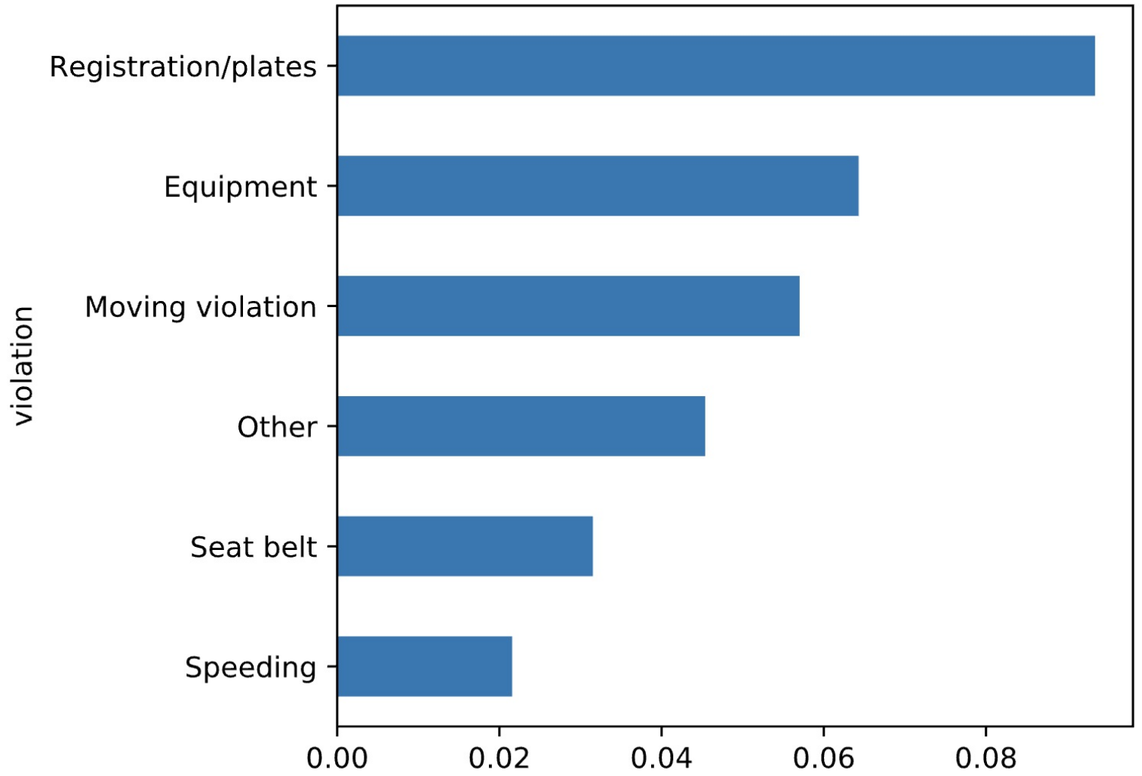
Ordering the bars (2)

```
search_rate.sort_values().plot(kind='bar')
plt.show()
```



Rotating the bars

```
search_rate.sort_values().plot(kind='barh')
plt.show()
```



ANALYZING POLICE ACTIVITY WITH PANDAS

Let's practice!