Indexing DataFrames

MANIPULATING DATAFRAMES WITH PANDAS

Anaconda
Instructor

A simple DataFrame

```python
import pandas as pd
df = pd.read_csv('sales.csv', index_col='month')
df
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
May     132   NaN    52
Jun     205  60.0    55
```

Indexing using square brackets

```
df
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
May     132   NaN    52
Jun     205  60.0    55
```

```
df['salt']['Jan']
```

```
12.0
```

Using column attribute and row label

```
df
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
May     132   NaN    52
Jun     205  60.0    55
```

```
df.eggs['Mar']
```

```
221
```

Using the .loc accessor

```
df
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
May     132   NaN    52
Jun     205  60.0    55
```

```
df.loc['May', 'spam']
```

```
52.0
```

Using the .iloc accessor

```
df
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
May     132   NaN    52
Jun     205  60.0    55
```

```
df.iloc[4, 2]
```

```
52.0
```

Selecting only some columns

```python
df_new = df[['salt','eggs']]
df_new
```

```
       salt  eggs
month
Jan    12.0    47
Feb    50.0   110
Mar    89.0   221
Apr    87.0    77
May     NaN   132
Jun    60.0   205
```

Let's practice!

MANIPULATING DATAFRAMES WITH PANDAS

Slicing DataFrames

MANIPULATING DATAFRAMES WITH PANDAS

Anaconda
Instructor

datacamp

sales DataFrame

```
df
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
May     132   NaN    52
Jun     205  60.0    55
```

Selecting a column (i.e., Series)

```
df['eggs']
```

```
month
Jan     47
Feb    110
Mar    221
Apr     77
May    132
Jun    205
Name: eggs, dtype: int64
```

```
type(df['eggs'])
```

```
pandas.core.series.Series
```

Slicing and indexing a Series

```
df['eggs'][1:4] # Part of the eggs column
```

```
month
Feb    110
Mar    221
Apr     77
Name: eggs, dtype: int64
```

```
df['eggs'][4]   # The value associated with May
```

```
132
```

Using .loc[]

```
df.loc[:, 'eggs':'salt'] # All rows, some columns
```

```
       eggs  salt
month
Jan      47  12.0
Feb     110  50.0
Mar     221  89.0
Apr      77  87.0
May     132   NaN
Jun     205  60.0
```

Using .loc[]

```
df.loc['Jan':'Apr',:] # Some rows, all columns
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
```

Using .loc[]

```
df.loc['Mar':'May', 'salt':'spam']
```

```
       salt  spam
month
Mar    89.0    72
Apr    87.0    20
May     NaN    52
```

Using .iloc[]

```
df.iloc[2:5, 1:] # A block from middle of the DataFrame
```

```
       salt  spam
month
Mar    89.0    72
Apr    87.0    20
May     NaN    52
```

Using lists rather than slices

```
df.loc['Jan':'May', ['eggs', 'spam']]
```

```
       eggs  spam
month
Jan      47    17
Feb     110    31
Mar     221    72
Apr      77    20
May     132    52
```

Using lists rather than slices

```
df.iloc[[0,4,5], 0:2]
```

```
       eggs  salt
month
Jan      47  12.0
May     132   NaN
Jun     205  60.0
```

Series versus 1-column DataFrame

```
# A Series by column name
df['eggs']
```

```
       eggs
month
Jan      47
Feb     110
Mar     221
...        ...
```

```
type(df['eggs'])
```

```
pandas.core.series.Series
```

```
# A DataFrame w/single column
df[['eggs']]
```

```
       eggs
month
Jan      47
Feb     110
Mar     221
...        ...
```

```
type(df[['eggs']])
```

```
pandas.core.frame.DataFrame
```

Let's practice!

MANIPULATING DATAFRAMES WITH PANDAS

Filtering DataFrames

MANIPULATING DATAFRAMES WITH PANDAS

Anaconda
Instructor

datacamp

Creating a Boolean Series

```
df.salt > 60
```

```
month
Jan    False
Feb    False
Mar     True
Apr     True
May    False
Jun    False
Name: salt, dtype: bool
```

Filtering with a Boolean Series

```
df[df.salt > 60]
```

```
       eggs  salt  spam
month
Mar     221  89.0    72
Apr      77  87.0    20
```

```
enough_salt_sold = df.salt > 60
df[enough_salt_sold]
```

```
       eggs  salt  spam
month
Mar     221  89.0    72
Apr      77  87.0    20
```

Combining filters

```
df[(df.salt >= 50) & (df.eggs < 200)] # Both conditions
```

```
       eggs  salt  spam
month
Feb     110  50.0    31
Apr      77  87.0    20
```

```
df[(df.salt >= 50) | (df.eggs < 200)] # Either condition
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
May     132   NaN    52
Jun     205  60.0    55
```

DataFrames with zeros and NaNs

```
df2 = df.copy()
df2['bacon'] = [0, 0, 50, 60, 70, 80]
df2
```

```
       eggs  salt  spam  bacon
month
Jan      47  12.0    17      0
Feb     110  50.0    31      0
Mar     221  89.0    72     50
Apr      77  87.0    20     60
May     132   NaN    52     70
Jun     205  60.0    55     80
```

Select columns with all nonzeros

```
df2.loc[:, df2.all()]
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
May     132   NaN    52
Jun     205  60.0    55
```

Select columns with any nonzeros

```
df2.loc[:, df2.any()]
```

```
       eggs  salt  spam  bacon
month
Jan      47  12.0    17      0
Feb     110  50.0    31      0
Mar     221  89.0    72     50
Apr      77  87.0    20     60
May     132   NaN    52     70
Jun     205  60.0    55     80
```

Select columns with any NaNs

```
df.loc[:, df.isnull().any()]
```

```
       salt
month
Jan    12.0
Feb    50.0
Mar    89.0
Apr    87.0
May     NaN
Jun    60.0
```

Select columns without NaNs

```
df.loc[:, df.notnull().all()]
```

```
       eggs  spam
month
Jan      47    17
Feb     110    31
Mar     221    72
Apr      77    20
May     132    52
Jun     205    55
```

Drop rows with any NaNs

```
df.dropna(how='any')
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     221  89.0    72
Apr      77  87.0    20
Jun     205  60.0    55
```

Filtering a column based on another

```
df.eggs[df.salt > 55]
```

```
month
Mar    221
Apr     77
Jun    205
Name: eggs, dtype: int64
```

Modifying a column based on another

```
df.eggs[df.salt > 55] += 5
df
```

```
       eggs  salt  spam
month
Jan      47  12.0    17
Feb     110  50.0    31
Mar     226  89.0    72
Apr      82  87.0    20
May     132   NaN    52
Jun     210  60.0    55
```

Let's practice!

MANIPULATING DATAFRAMES WITH PANDAS

Transforming DataFrames

MANIPULATING DATAFRAMES WITH PANDAS

Anaconda
Instructor

DataFrame vectorized methods

```
df.floordiv(12)  # Convert to dozens unit
```

```
       eggs  salt  spam
month
Jan       3   1.0     1
Feb       9   4.0     2
Mar      18   7.0     6
Apr       6   7.0     1
May      11   NaN     4
Jun      17   5.0     4
```

NumPy vectorized functions

```python
import numpy as np
np.floor_divide(df, 12)  # Convert to dozens unit
```

```
       eggs  salt  spam
month
Jan     3.0   1.0   1.0
Feb     9.0   4.0   2.0
Mar    18.0   7.0   6.0
Apr     6.0   7.0   1.0
May    11.0   NaN   4.0
Jun    17.0   5.0   4.0
```

Plain Python functions

```
def dozens(n):
    return n // 12
```

```
df.apply(dozens)  # Convert to dozens unit
```

```
       eggs  salt  spam
month
Jan       3   1.0     1
Feb       9   4.0     2
Mar      18   7.0     6
Apr       6   7.0     1
May      11   NaN     4
Jun      17   5.0     4
```

Plain Python functions

```
df.apply(lambda n: n // 12)
```

```
       eggs  salt  spam
month
Jan       3   1.0     1
Feb       9   4.0     2
Mar      18   7.0     6
Apr       6   7.0     1
May      11   NaN     4
Jun      17   5.0     4
```

Storing a transformation

```
df['dozens_of_eggs'] = df.eggs.floordiv(12)
df
```

```
       eggs  salt  spam  dozens_of_eggs
month
Jan      47  12.0    17               3
Feb     110  50.0    31               9
Mar     221  89.0    72              18
Apr      77  87.0    20               6
May     132   NaN    52              11
Jun     205  60.0    55              17
```

The DataFrame index

```
df
```

```
       eggs  salt  spam  dozens_of_eggs
month
Jan      47  12.0    17               3
Feb     110  50.0    31               9
Mar     221  89.0    72              18
Apr      77  87.0    20               6
May     132   NaN    52              11
Jun     205  60.0    55              17
```

```
df.index
```

```
Index(['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun'], dtype='object',
name='month')
```

Working with string values

```
df.index = df.index.str.upper()
df
```

```
       eggs  salt  spam  dozens_of_eggs
month
JAN      47  12.0    17               3
FEB     110  50.0    31               9
MAR     221  89.0    72              18
APR      77  87.0    20               6
MAY     132   NaN    52              11
JUN     205  60.0    55              17
```

Working with string values

```
df.index = df.index.map(str.lower)
df
```

```
     eggs  salt  spam  dozens_of_eggs
jan    47  12.0    17               3
feb   110  50.0    31               9
mar   221  89.0    72              18
apr    77  87.0    20               6
may   132   NaN    52              11
jun   205  60.0    55              17
```

Defining columns using other columns

```python
df['salty_eggs'] = df.salt + df.dozens_of_eggs
df
```

```
     eggs  salt  spam  dozens_of_eggs  salty_eggs
jan    47  12.0    17               3        15.0
feb   110  50.0    31               9        59.0
mar   221  89.0    72              18       107.0
apr    77  87.0    20               6        93.0
may   132   NaN    52              11         NaN
jun   205  60.0    55              17        77.0
```

Let's practice!

MANIPULATING DATAFRAMES WITH PANDAS