Data Handling with Pandas

•••

Essential python libraries

NumPy

Pandas

Matplotlib

Jupyter

SciPy

Scikit-learn

Statsmodels

Computing libraries in python

01 **Pandas**

- Data structures
- Tools

02 **Numpy**

- Arrays
- Matrices

03 **SciPy**

- Solving differential equations
- Optimization
- Data visualization

Pandas

pandas provides high-level data structures and functions designed to make working with structured or tabular data fast, easy, and expressive.

- DataFrame: a tabular, column-oriented data structure with both row and column labels
- Series: a one-dimensional labeled array object

Numpy

- A fast and efficient multidimensional array object ndarray
- Tools for reading and writing array-based datasets to disk
- Linear algebra operations, Fourier transform, and random number generation

Scipy

SciPy is a collection of packages addressing a number of different standard problem domains in scientific computing.

example)

scipy.stats

Standard continuous and discrete probability distributions (density functions, samplers, continuous distribution functions), various statistical tests, and more descriptive statistics

Visualization libraries

1	Matplotlib	● Plots ● Graphs ● Most popular
2	Seaborn	● Heat maps ● Time series ● Violin plots

Algorithmic libraries

1	Scikit-learn	● Machine learning ● Regression ● Classification
2	Statsmodels	● Explore data ● Estimate statistical model ● Perform statistical tests

Scikit-learn

scikit-learn has become the premier general-purpose machine learning toolkit for Python programmers.

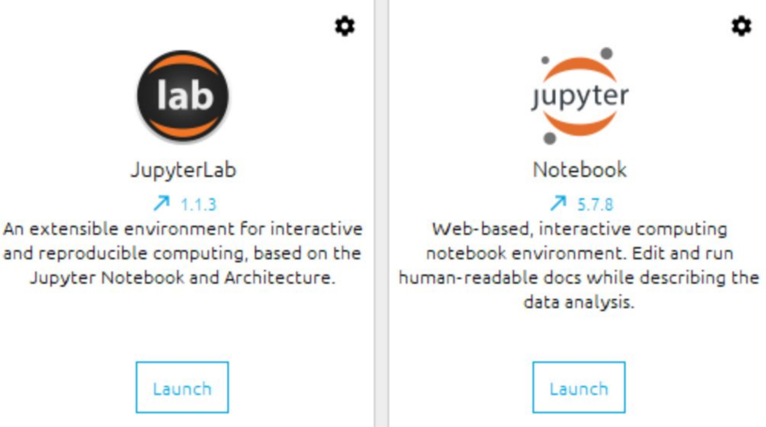
Statsmodels

Statsmodels is also a Python module that allows users to explore data, estimate statistical models and perform statistical tests

Compared with scikit-learn, statsmodels contains algorithms for classical (primarily frequentist) statistics and econometrics.

Jupyter

The Jupyter notebook system also allows you to author content in Markdown and HTML, providing you a means to create rich documents with code and text.



JupyterLab

↗ 1.1.3

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook

↗ 5.7.8

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Importing and exploring data in python

Importing data

Q. What is importing data?

Process of getting the data into python

Two important properties to import data into python

1) Format (e.g., .csv, .json, .xlsx)
2) File path of dataset (e.g., /desktop/demo.csv)

Importing a CSV into python

To use pandas, let’s type the line below

```
Import pandas as pd
```

```
File_path = “        “
```

→ define a variable with a file path

```
Df = pd.read_csv (file_path)
```

→ in pandas, the read_CSV can read in files with column separated by commas into a pandas dataframe

Importing a CSV into python without a header

```
Import pandas as pd

File_path = “       “

Df = pd.read_csv (file_path, header = none)
```

Creating data

There are two core objects in pandas: the DataFrame and the Series.

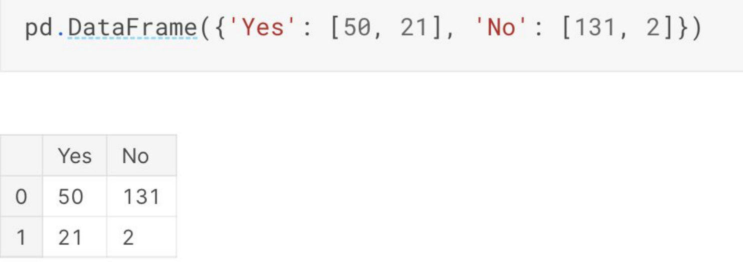
DataFrame

A DataFrame is a table. It contains an array of individual entries, each of which has a certain value.

Creating data

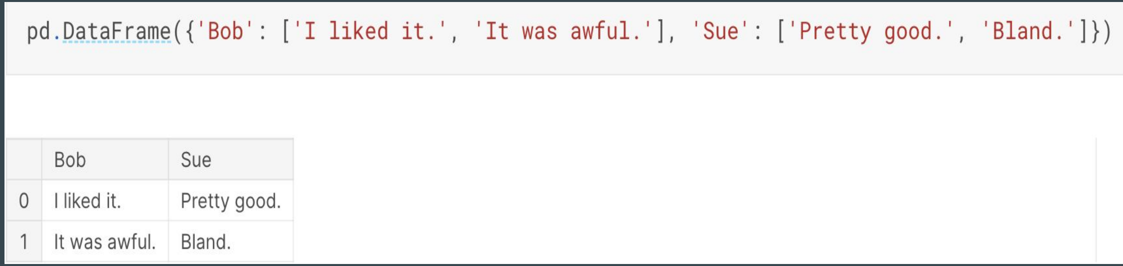
DataFrame

A DataFrame is a table. It contains an array of individual entries, each of which has a certain value.



```
pd.DataFrame({'Yes': [50, 21], 'No': [131, 2]})
```

	Yes	No
0	50	131
1	21	2



```
pd.DataFrame({'Bob': ['I liked it.', 'It was awful.'], 'Sue': ['Pretty good.', 'Bland.']})
```

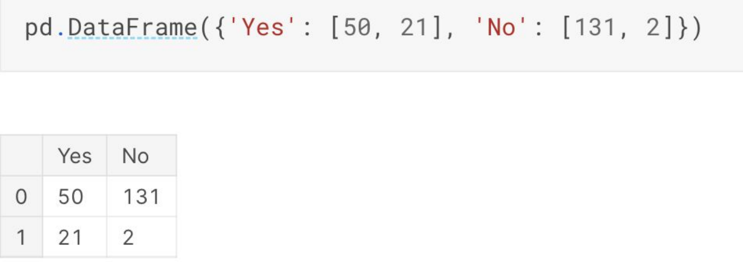
	Bob	Sue
0	I liked it.	Pretty good.
1	It was awful.	Bland.

DataFrame entries are not limited to integers

Creating data

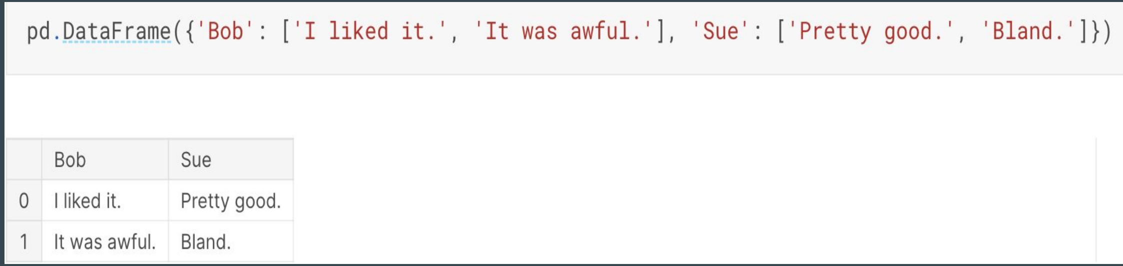
DataFrame

pd.DataFrame () : generate dataframe objects. The syntax for declaring a new one is a dictionary whose keys are the column names



```python
pd.DataFrame({'Yes': [50, 21], 'No': [131, 2]})
```

	Yes	No
0	50	131
1	21	2



```python
pd.DataFrame({'Bob': ['I liked it.', 'It was awful.'], 'Sue': ['Pretty good.', 'Bland.']})
```

	Bob	Sue
0	I liked it.	Pretty good.
1	It was awful.	Bland.

Printing the dataframe in Python

df prints the whole dataframe → can take a lot of time for big datasets

df.head(n): shows the *first* n rows of dataframe

df.tail(n): shows the *last* n rows of dataframe

df.shape: check how large the resulting dataframe is

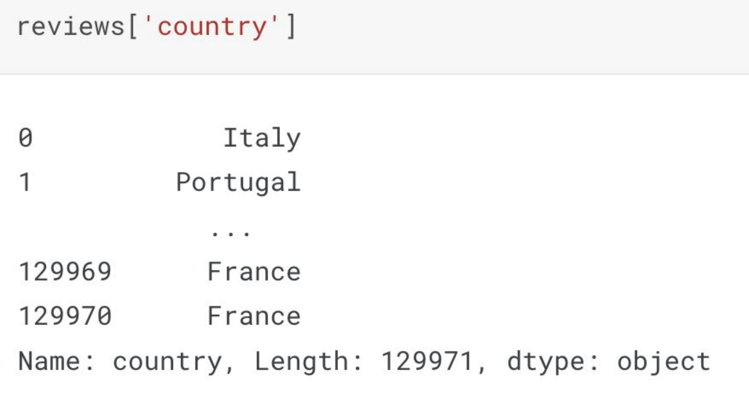
Exporting to different formats in python

Data format	Read	save
csv	pd.read_csv()	df.to_csv()
json	pd.read_json()	df.to_json()
excel	pd.read_excel()	df.to_excel()
sql	pd.read_sql()	df.to_sql()

Indexing

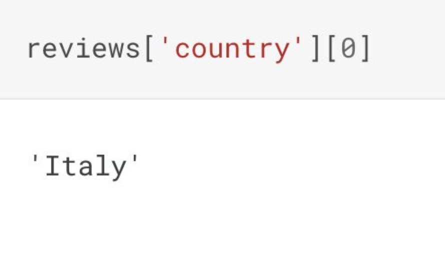
How to go about selecting the data points relevant to you quickly and effectively.

If we have a Python dictionary, we can access its values using the indexing ([]) operator. We can do the same with columns in a DataFrame:



```
reviews['country']
```

```
0            Italy
1         Portugal
            ...
129969      France
129970      France
Name: country, Length: 129971, dtype: object
```



```
reviews['country'][0]
```

```
'Italy'
```

Indexing in pandas

pandas has its own accessor operators, loc and iloc

Index-based selection

Pandas indexing works in one of two paradigms. The first is index-based selection: selecting data based on its numerical position in the data. **iloc** follows this paradigm

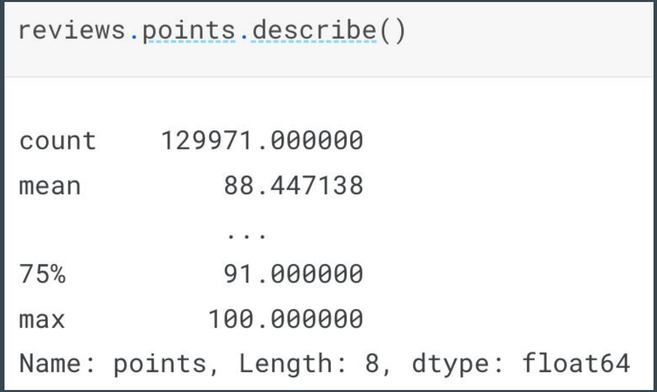
Label-based selection

The second paradigm for attribute selection is the one followed by the loc operator: label-based selection. In this paradigm, it's the data index value, not its position, which matters.

Summary functions

Pandas provides many simple "summary functions" (not an official name) which restructure the data in some useful way

Describe () method: This method generates a high-level summary of the attributes of the given column.



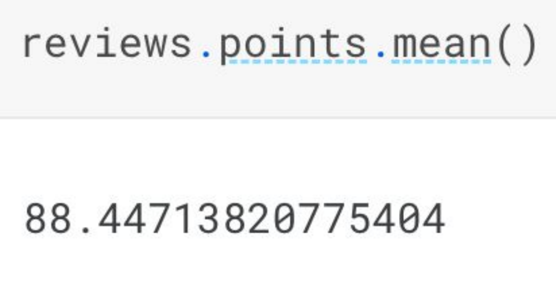
```
reviews.points.describe()
```

```
count    129971.000000
mean         88.447138
                 ...
75%          91.000000
max         100.000000
Name: points, Length: 8, dtype: float64
```

Summary functions

If you want to get some particular simple summary statistic about a column in a DataFrame or a Series, there is usually a helpful pandas function that makes it happen

Mean () functions



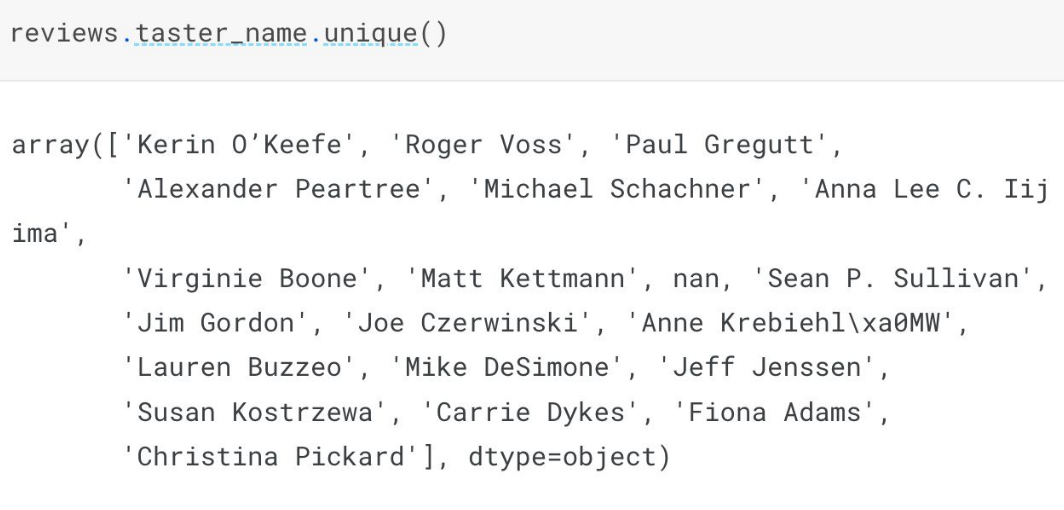
```
reviews.points.mean()
```

```
88.44713820775404
```

Summary functions

To see a list of unique values, we can use the

unique () functions



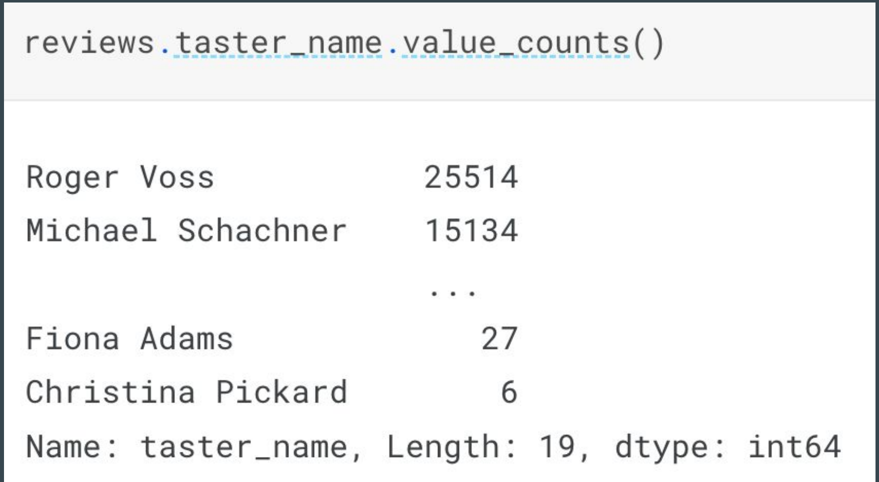
```
reviews.taster_name.unique()
```

```
array(['Kerin O’Keefe', 'Roger Voss', 'Paul Gregutt',
       'Alexander Peartree', 'Michael Schachner', 'Anna Lee C. Iij
ima',
       'Virginie Boone', 'Matt Kettmann', nan, 'Sean P. Sullivan',
       'Jim Gordon', 'Joe Czerwinski', 'Anne Krebiehl\xa0MW',
       'Lauren Buzzeo', 'Mike DeSimone', 'Jeff Jenssen',
       'Susan Kostrzewa', 'Carrie Dykes', 'Fiona Adams',
       'Christina Pickard'], dtype=object)
```

Summary functions

To see a list of unique values and how often they occur in the dataset, we can use the

values_counts () functions



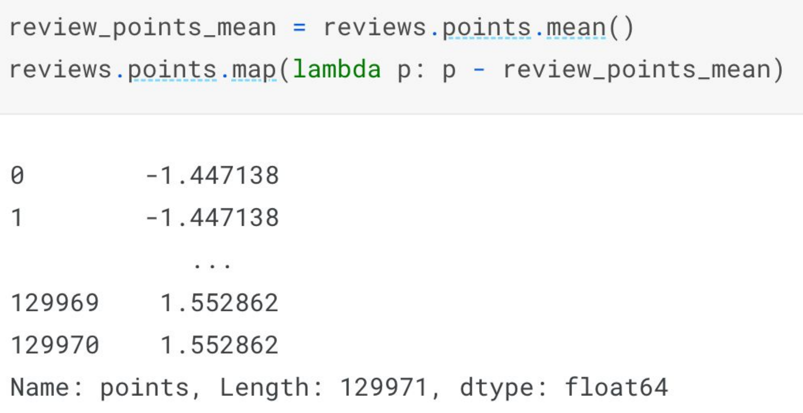
```
reviews.taster_name.value_counts()
```

```
Roger Voss            25514
Michael Schachner     15134
                        ...
Fiona Adams              27
Christina Pickard         6
Name: taster_name, Length: 19, dtype: int64
```

Maps

Need for creating new representations from existing data, or for transforming data from the format it is in now to the format that we want it to be

1) Map ()



```
review_points_mean = reviews.points.mean()
reviews.points.map(lambda p: p - review_points_mean)
```

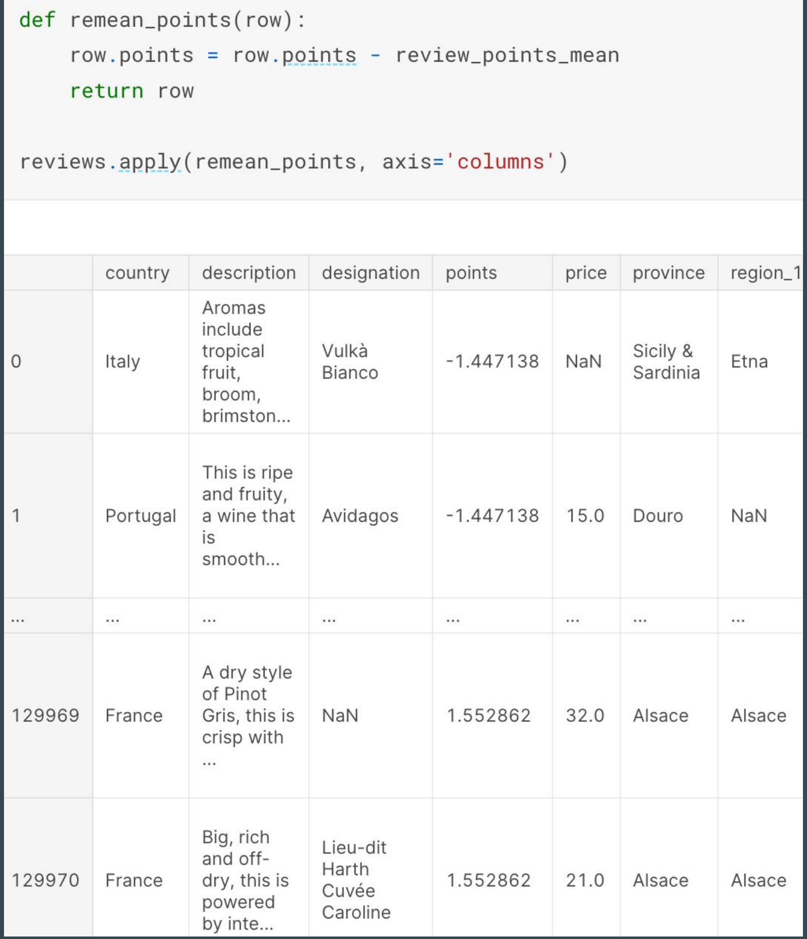
```
0         -1.447138
1         -1.447138
             ...
129969     1.552862
129970     1.552862
Name: points, Length: 129971, dtype: float64
```

Maps

Need for creating new representations from existing data, or for transforming data from the format it is in now to the format that we want it to be

2) Apply ()

the equivalent method if we want to transform a whole DataFrame by calling a custom method on each row.



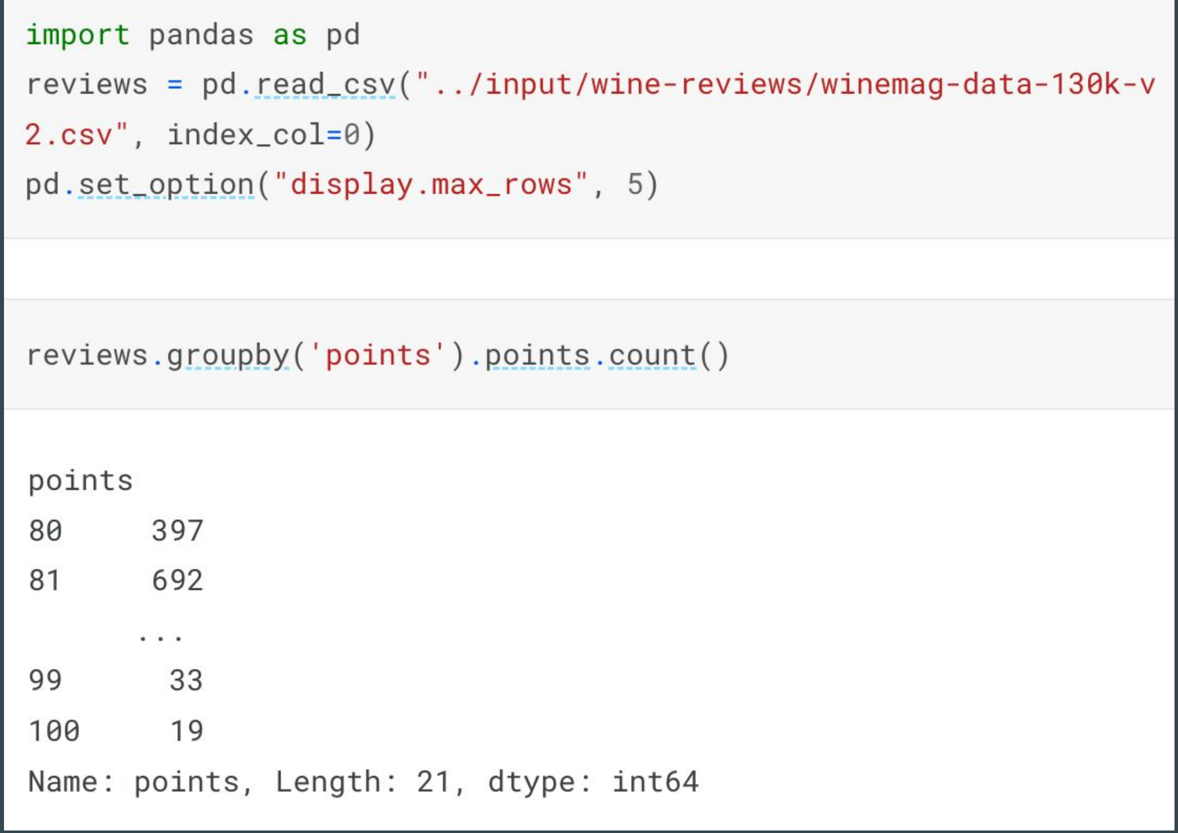
```
def remean_points(row):
    row.points = row.points - review_points_mean
    return row

reviews.apply(remean_points, axis='columns')
```

	country	description	designation	points	price	province	region_1
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	-1.447138	NaN	Sicily & Sardinia	Etna
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	-1.447138	15.0	Douro	NaN
...	...	...	...	...	...	...	...
129969	France	A dry style of Pinot Gris, this is crisp with ...	NaN	1.552862	32.0	Alsace	Alsace
129970	France	Big, rich and off-dry, this is powered by inte...	Lieu-dit Harth Cuvée Caroline	1.552862	21.0	Alsace	Alsace

Grouping and sorting

Groupby ()



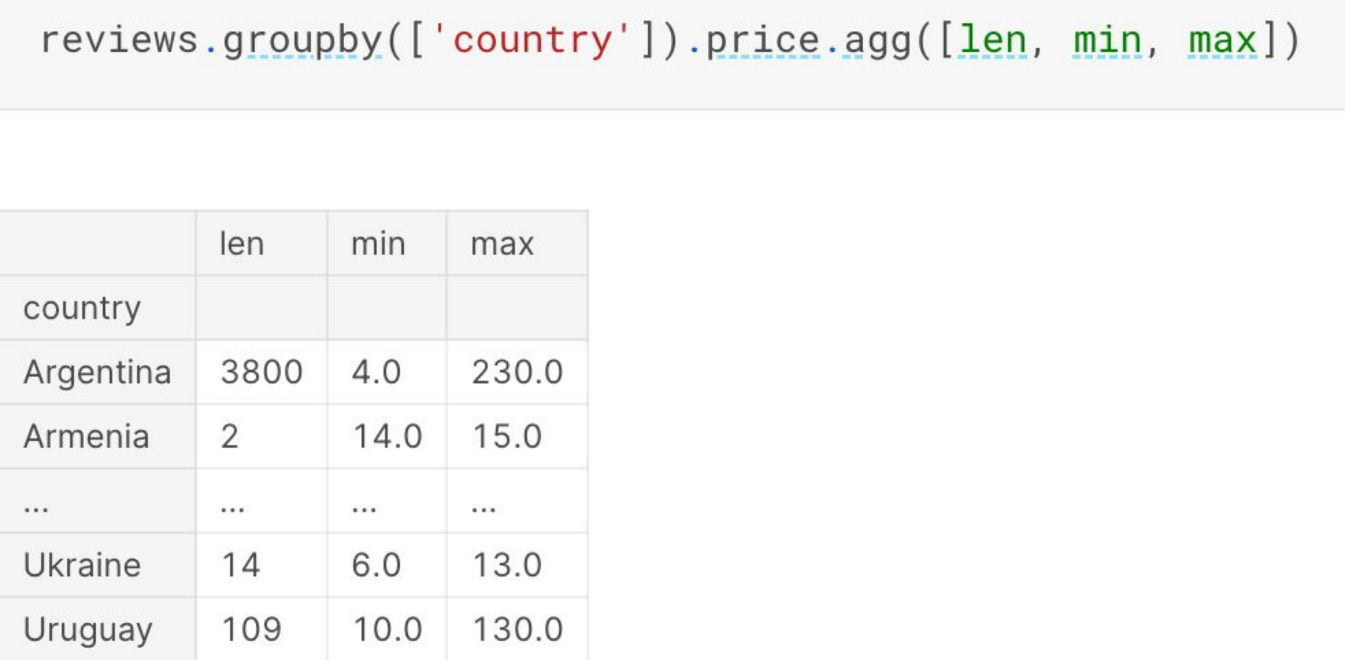
```
import pandas as pd
reviews = pd.read_csv("../input/wine-reviews/winemag-data-130k-v
2.csv", index_col=0)
pd.set_option("display.max_rows", 5)
```

```
reviews.groupby('points').points.count()
```

```
points
80      397
81      692
       ...
99       33
100      19
Name: points, Length: 21, dtype: int64
```

Grouping and sorting

Agg ()

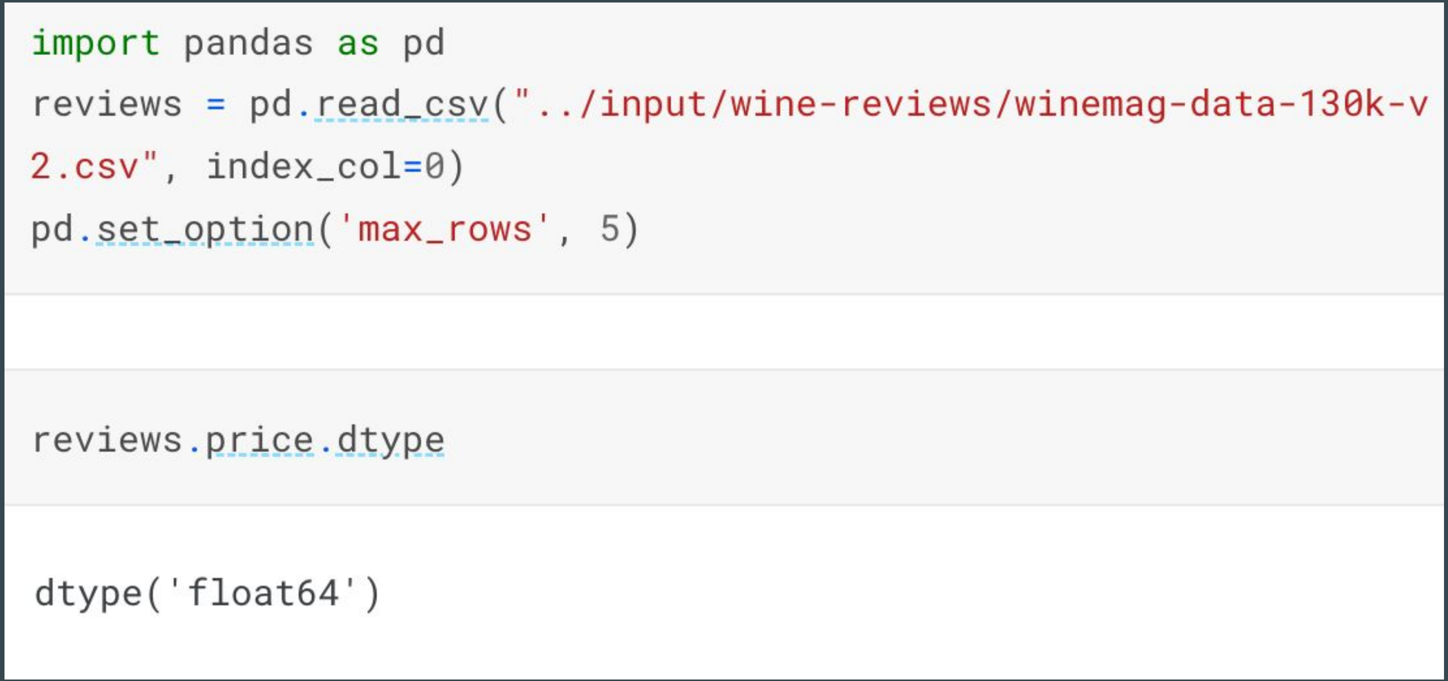


```
reviews.groupby(['country']).price.agg([len, min, max])
```

	len	min	max
country			
Argentina	3800	4.0	230.0
Armenia	2	14.0	15.0
...	...	...	...
Ukraine	14	6.0	13.0
Uruguay	109	10.0	130.0

Data types

The data type for a column in a DataFrame or a Series is known as the dtype.



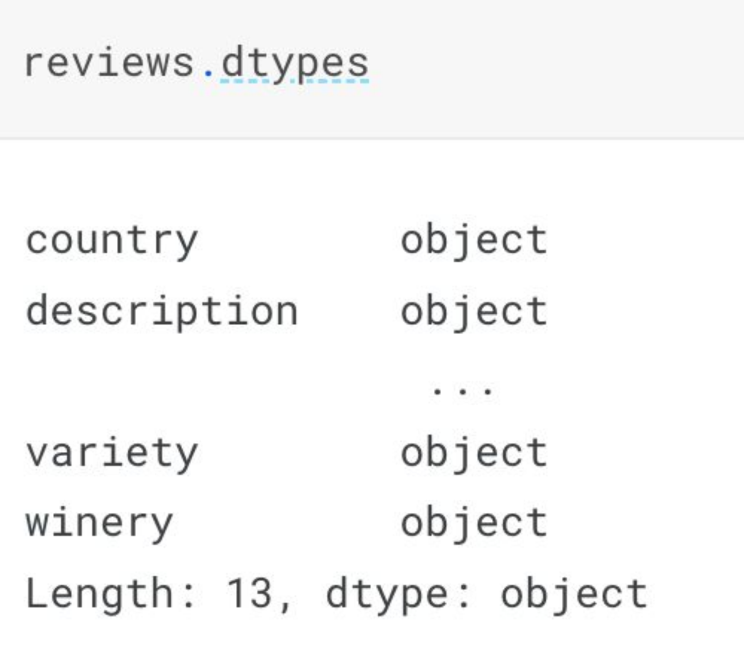
```
import pandas as pd
reviews = pd.read_csv("../input/wine-reviews/winemag-data-130k-v
2.csv", index_col=0)
pd.set_option('max_rows', 5)
```

```
reviews.price.dtype
```

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

Data types

the dtypes property returns the dtype of every column in the DataFrame:



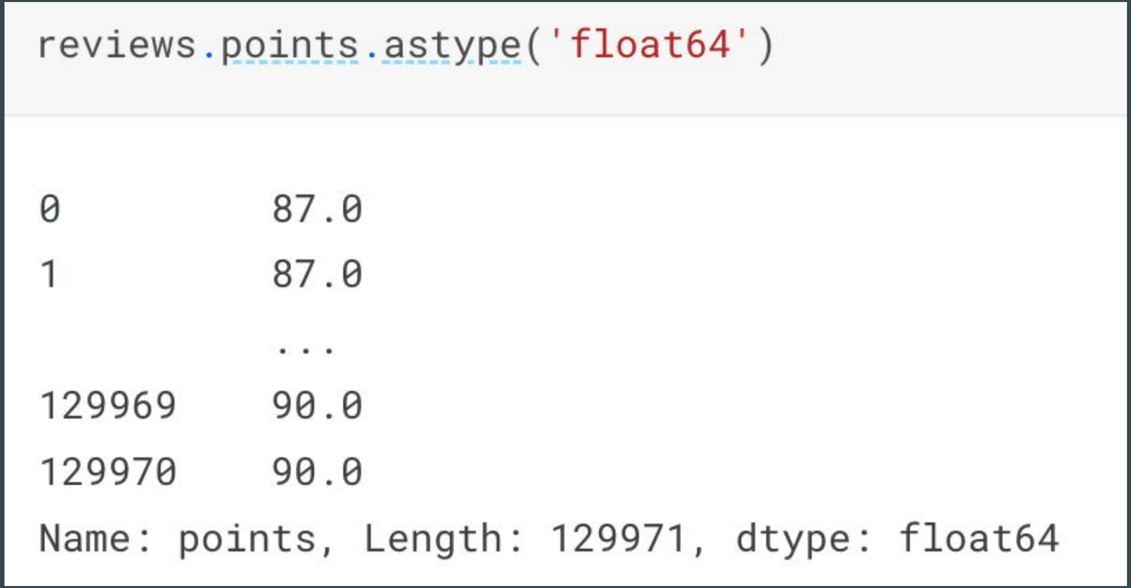
```
reviews.dtypes
```

```
country          object
description      object
                   ...
variety          object
winery           object
Length: 13, dtype: object
```

Data types

convert a column of one type into another

astype() function



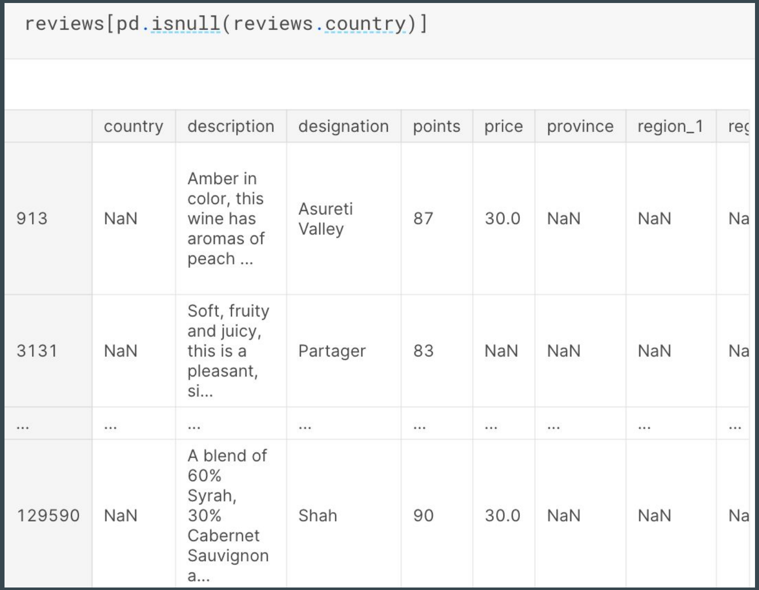
```
reviews.points.astype('float64')
```

```
0         87.0
1         87.0
            ...
129969    90.0
129970    90.0
Name: points, Length: 129971, dtype: float64
```

Missing data

Entries missing values are given the value NaN, short for "Not a Number". For technical reasons these NaN values are always of the float64 dtype

How to select missing data in pandas? pd.isnull()

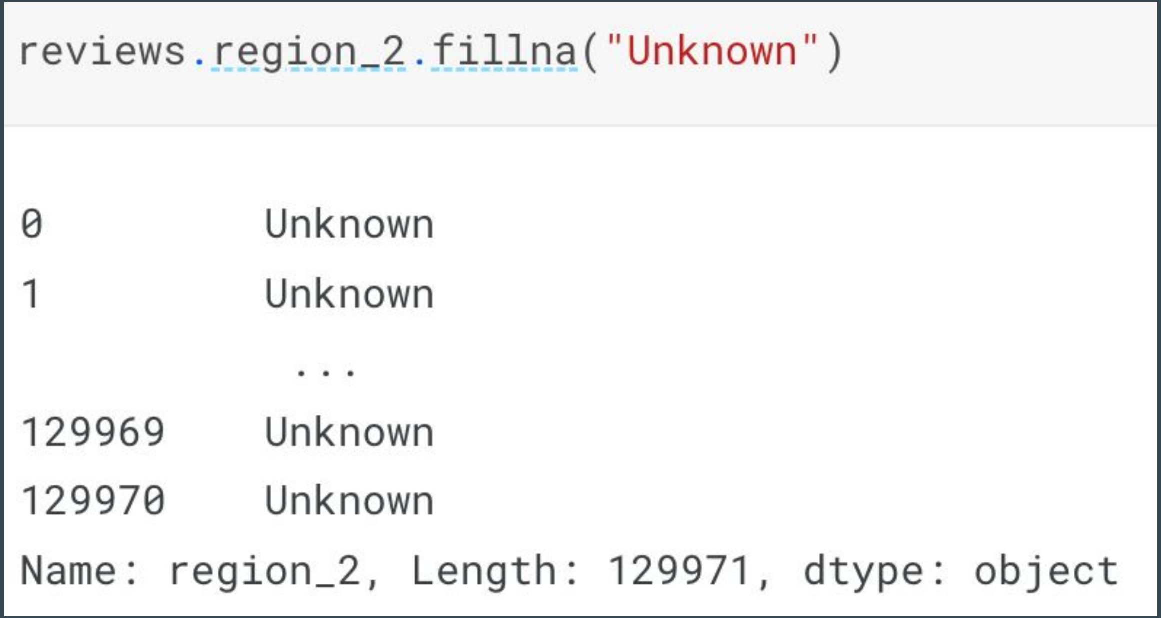


```
reviews[pd.isnull(reviews.country)]
```

	country	description	designation	points	price	province	region_1	reg
913	NaN	Amber in color, this wine has aromas of peach ...	Asureti Valley	87	30.0	NaN	NaN	Na
3131	NaN	Soft, fruity and juicy, this is a pleasant, si...	Partager	83	NaN	NaN	NaN	Na
...	...	...	...	...	...	...	...	...
129590	NaN	A blend of 60% Syrah, 30% Cabernet Sauvignon a...	Shah	90	30.0	NaN	NaN	Na

Missing data

Replacing missing value: fillna()

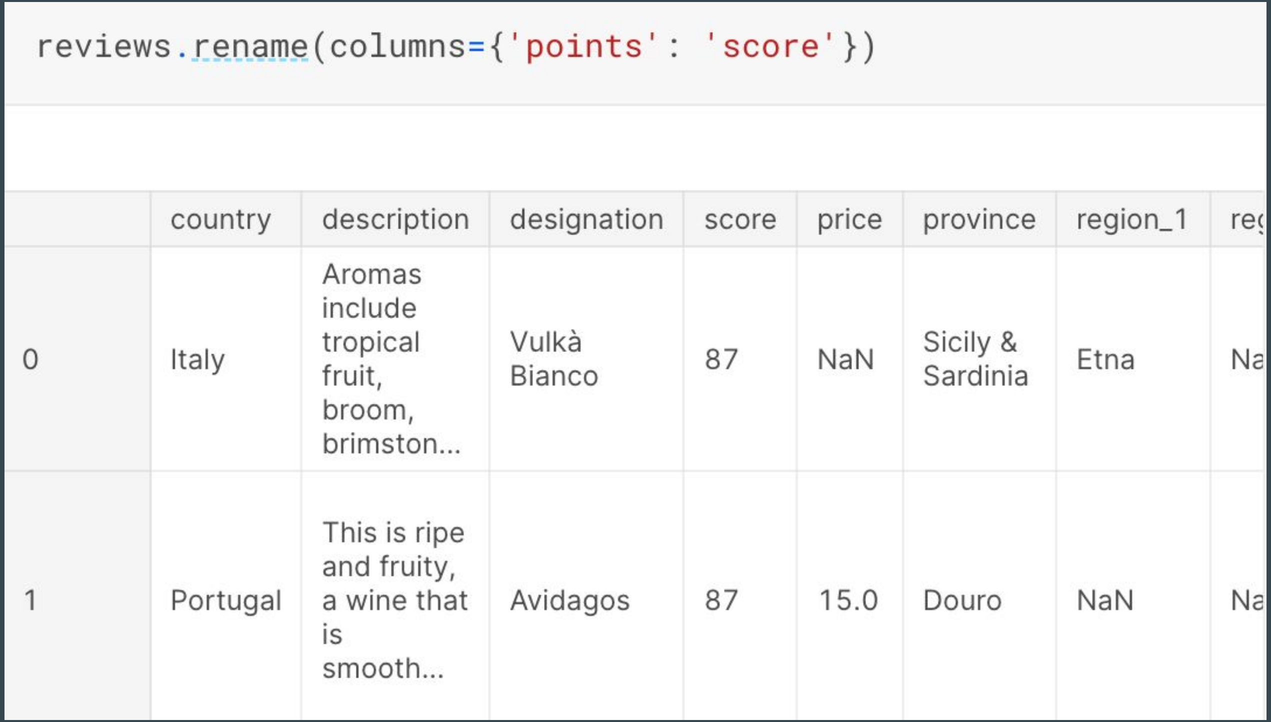


```
reviews.region_2.fillna("Unknown")
```

```
0         Unknown
1         Unknown
            ...
129969    Unknown
129970    Unknown
Name: region_2, Length: 129971, dtype: object
```

Renaming

rename() function lets you change index names and/or column names

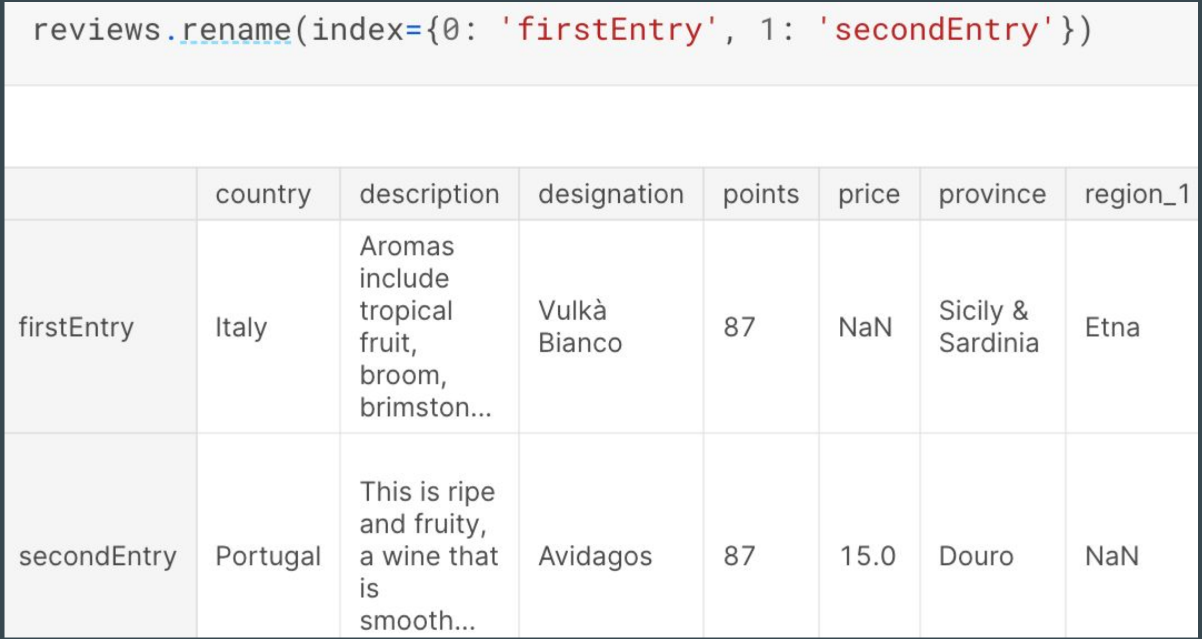


```
reviews.rename(columns={'points': 'score'})
```

	country	description	designation	score	price	province	region_1	reg
0	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna	Na
1	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN	Na

Renaming

rename() lets you rename index or column values by specifying a index or column keyword parameter, respectively

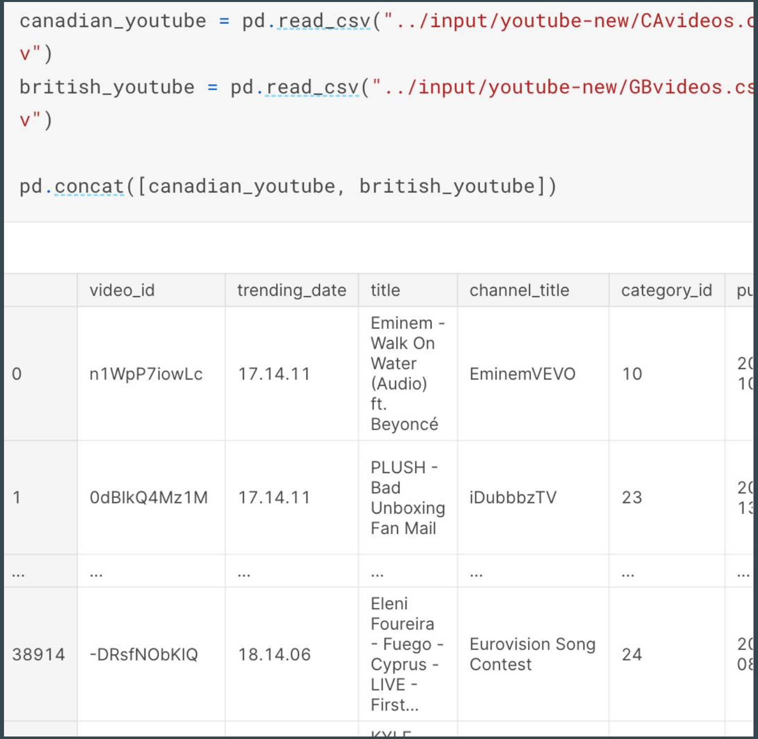


```
reviews.rename(index={0: 'firstEntry', 1: 'secondEntry'})
```

	country	description	designation	points	price	province	region_1
firstEntry	Italy	Aromas include tropical fruit, broom, brimston...	Vulkà Bianco	87	NaN	Sicily & Sardinia	Etna
secondEntry	Portugal	This is ripe and fruity, a wine that is smooth...	Avidagos	87	15.0	Douro	NaN

Combining

1) concat (): Simplest combining function



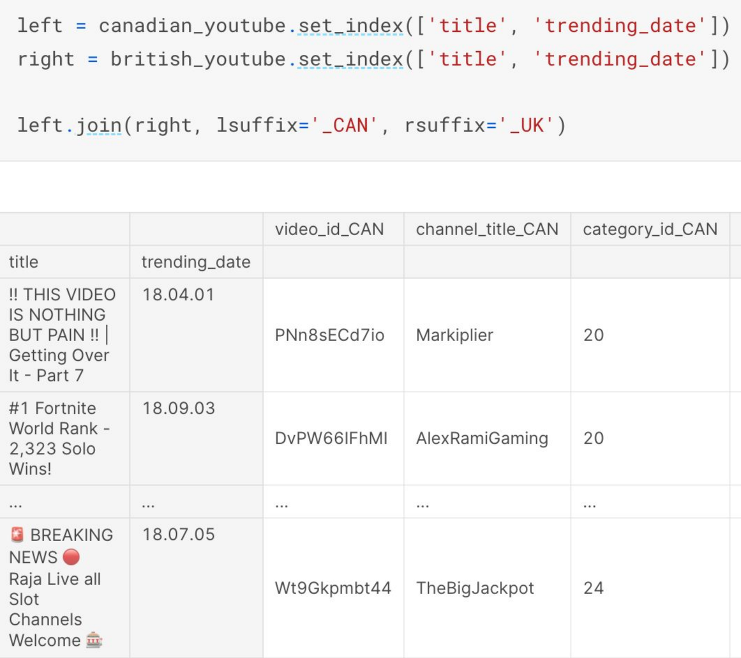
```
canadian_youtube = pd.read_csv("../input/youtube-new/CAvideos.c
v")
british_youtube = pd.read_csv("../input/youtube-new/GBvideos.cs
v")

pd.concat([canadian_youtube, british_youtube])
```

	video_id	trending_date	title	channel_title	category_id	pu
0	n1WpP7iowLc	17.14.11	Eminem - Walk On Water (Audio) ft. Beyoncé	EminemVEVO	10	20 10
1	0dBIkQ4Mz1M	17.14.11	PLUSH - Bad Unboxing Fan Mail	iDubbbzTV	23	20 13
...	...	...	...	...	...	...
38914	-DRsfNObKIQ	18.14.06	Eleni Foureira - Fuego - Cyprus - LIVE - First...	Eurovision Song Contest	24	20 08

Combining

2) join (): lets you combine different DataFrame objects which have an index in common



```
left = canadian_youtube.set_index(['title', 'trending_date'])
right = british_youtube.set_index(['title', 'trending_date'])

left.join(right, lsuffix='_CAN', rsuffix='_UK')
```

		video_id_CAN	channel_title_CAN	category_id_CAN
title	trending_date			
!! THIS VIDEO IS NOTHING BUT PAIN !! | Getting Over It - Part 7	18.04.01	PNn8sECd7io	Markiplier	20
#1 Fortnite World Rank - 2,323 Solo Wins!	18.09.03	DvPW66IFhMI	AlexRamiGaming	20
...	...	...	...	...
🚨 BREAKING NEWS 🔴 Raja Live all Slot Channels Welcome 🎰	18.07.05	Wt9Gkpmbt44	TheBigJackpot	24