



# PANDAS Data Science

Cheat Sheet



#### **PANDAS**

Pandas is one of the most popular Python libraries for data science and analytics. It helps you manage two-dimensional data tables and other data structures. It relies on Numpy, so when you import Pandas, you need to import Numpy first.

import numpy as np import pandas as pd

#### PANDAS DATA STRUCTURES

**Series:** Pandas Series is a one dimensional data structure ("a one dimensional ndarray") that can store values, with a unique index for each value.

```
in [4]:
            test_set_series
out [4]:
                             14
                             69
                  5
                             73
                             92
                             56
                  8
                           101
                  9
                           120
                 10
                           175
                 11
                           191
                 12
                           215
                 13
                           306
                 14
                           241
                 15
                           392
                 dtype: int64
```





**DataFrame:** Pandas DataFrame is a two (or more) dimensional data structure – basically a table with rows and columns. The columns have names and the rows have indexes.



in [12]:	b	ig_table				
out [12]:		user_id	phone_type	source	free	super
	0	1000001	android	invite_a_friend	5.0	0.0
	1	1000002	ios	invite_a_friend	4.0	0.0
	2	1000003	error	invite_a_friend	37.0	0.0
	3	1000004	error	invite_a_friend	0.0	0.0
	4	1000005	ios	invite_a_friend	6.0	0.0

#### **OPENING A .CSV FILE IN PANDAS**

pd.read\_csv('/home/your/folder/file.csv', delimiter=';')

This opens the .csv file that's located in /home/your/folder and called file.csv. The fields in the file are separated with semicolons (;).

df = pd.read\_csv('/home/your/folder/file.csv', delimiter=';')

This opens a .csv file and stores the output into a variable called df. (The variable name can be anything else - not just df.)

pd.read\_csv('file.csv', delimiter=';', names = ['column1', 'column2', 'column3'])
This opens file.csv. The fields in the file are separated with semicolons (;). We change the original names of the columns and set them to: 'column1', 'column2' and 'column3'.





### QUERYING DATA FROM PANDAS DATAFRAMES



#### df

It returns the whole dataframe. (Note: remember, when we opened the .csv file, we stored our dataframe into the **df** variable!)

#### df.head()

It returns the first 5 rows of df.

#### df.tail()

It returns the last 5 rows of df.

#### df.sample(7)

It returns 7 random rows from df.

#### df[['column1', 'column2']]

It returns column1 and column2 from df. (The output is in DataFrame format.)

#### df.column1

It returns column1 from **df**. (The output is in Series format.)

#### df[my\_dataframe.column1 == 'given\_value']

It returns all columns, but only those rows in which the value in column1 is 'given\_value'. (The output is in DataFrame format.)

#### df[['column1']][my\_dataframe.column1 == 'given\_value'].head()

It takes the column1 column — and only those rows in which the value in column1 is 'given\_value' — and returns only the first 5 rows. (The point is: you can combine things!)





#### **AGGREGATING IN PANDAS**

The most important pandas aggregate functions:

- .count()
- .sum()
- · .mean()
- .median()
- .max()
- .min()

#### Examples:

df.count()

It counts the number of rows in each column of df.

#### df.max()

It returns the maximum value from each column of df.

#### df.column1.max()

It returns the maximum value only from the column1 column of df.

#### **PANDAS GROUP BY**

The .groupby() operation is usually used with an aggregate function(.count(), .sum(), .mean(), .median(), etc.). It groups the rows by a given column's values. (The column is specified as the argument of the .groupby() operation.) Then we can calculate the aggregate for each group and get that returned to the screen.

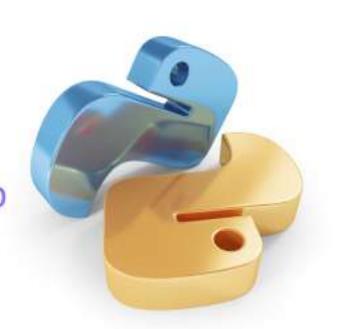






#### df.groupby('column1').count()

It counts the number of values in each column - for each group of unique column1 values.



#### df.groupby('column1').sum()

It sums the values in each column - for each group of unique column1 values.

#### df.groupby('column1').min()

It finds the minimum value in each column - for each group of unique column1 values.

#### df.groupby('column1').max()

It finds the maximum value in each column - for each group of unique column1 values.

#### A FEW MORE USEFUL PANDAS METHODS

#### df.merge(other\_df)

It joins df and other\_df - for every row where the value of column1 from df equals the value of column1 from other\_df.

#### df.sort\_values('column1')

It returns every row and column from **df**, sorted by column1, in ascending order (by default).

#### df.sort\_values('column1', ascending = False)

It returns every row and column from **df**, sorted by column1, in descending order.

#### df.fillna('some\_value')

It finds all empty (NaN) values in df and replaces them with 'some\_value'.





## FOLLOW US FOR MORE SUCH CONTENT

