
PANDAS
Library

Pandas

import Pandas as Pd

pd is object

pd.methods() — We gonna use for

To read a any file: csv, json, sqwl

csv — pd.read_csv("path")

json — pd.read_json("path")

sqwl — Pd.read_sqwl(query, connection)

To know the type

df = pd.read_csv("data.csv")

type(df) = dataframe

In excel, row and colums are called series.

Accessing any columns:-

df["City"] -> type(df["City"]) -> Series

This will shows the city rows data

df.City -> type(df.city) -> Series

Skip the row:-

df.read_csv("data.csv", skiprow=4

(4 rows are skipped)

Demension of dataframe

df.shape => (Total row, total clm)

df.shape[0] = column

first and last top results of dataframe:-

-> df.head() -> top few data

-> df.tail() -> last few data

Dataframe.information

df.info() — returns information about data.

value_counts()

gives the analysis of most occuring data frequencies.

df.city.value_counts()

ascending = True / False
dropna = drop empty values.

sort_values()

→ gives the sorted Series
→ sort the multiple series at a time. sort_values(by=)

new series = df.city.sort_values()

↙

Because its inplace = false means it will create a new series instead of overwriting the first one.

Boolean Indexing:-

AND = &
OR = |
NOT = ~

df.Medal == 'GOLD' → returns True/False

df[df.Medal = "GOLD"] → returns dataframe which have GOLD

df[(df.Medal = "GOLD") & (df.City = "Delhi")]

This returns dataframe which consists having Gold and living in Delhi

String handling:- (Str.contains(""))

→ df.name.str.contains("Krishna")

→ returns true/false in all dataframe

→ df[df.name.str.contains("Krishna")]

→ returns dataframe which contains Krishna

challenge

1) In which events did Jesse Owens win a medal?

Sol-

```
df[(df.Athletes.str.contains("Jesse Owens")) &
   (df.Medal == "GOLD" or
    df.Medal == "Silver" or
    df.Medal == "Bronze")]
```

OR

```
New = df[df.Athlete == "Jesse Owens"]
```

→ give all Jesse Owens Events stored in new variable

→ new.events.value_counts()

→ give all events.

2) Most Men Gold medal in Swimming Sort by Player name.

-> df[(df.Medal == "Gold") & (df.Gender == "Men") & (df.sport = "Swimming")]

-> Store in variable.

=> new.sort_values(by = "Athlete")

3) Which three countries won the most medals in recent years (1984 to 2008)

new = df(df.years >= 1984)

new.NOC.value_counts().head(3)

give result of three country which high medals.

4) Display male gold medal for 100m from most recent with City, Edition, Athlete and country.

```
new = df[(df.gender = "male") & (df.medal = "Gold") & (df.sport == "100m")]
new.sort_values('Edition", ascending = False)
```

to print extra series this gives

[['City', 'Edition', 'Athlete', 'NOC']]

BASIC PLOTTING

-> Matplotlib

Import matplotlib.pyplot as plt
% matplotlib inline

-> merger command -> helps to view graphs in notebook

Plot types -> Plot()

-> plot(kind = 'line') - Shortcut time period
-> plot(kind = 'bar') -> change or large
-> plot(kind = 'barh') - some
-> plot(kind = 'pie') -> show categories

```
fo = df[df.event == 1896]
fo.head(5)
fo.sport.value_counts().plot();
```

-> to remove the object line

default = line

color

```
fo.sport.value_count().plot(kind='bar',
                            color='maroon')
```

figsize

```
plot(figsize=(width, height))
fo.sport.value_count().plot(kind='bar',
                            figsize=(10,3))
```

countplot() — for complex data

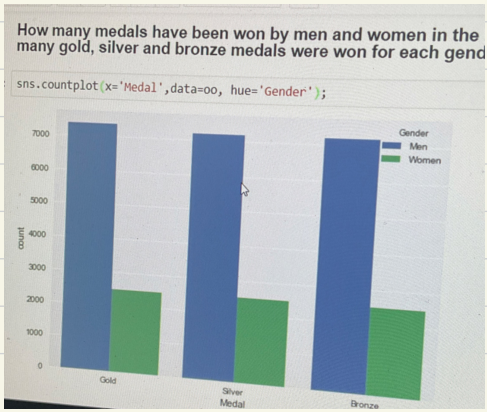
```
seaborn.countplot( ):
```

data - the dataframe
hue - categorical variables
order - sequence when using categorical level
palette - colors to use for the different levels of hue variable

import seaborn as sns

sns.countplot(x='medal', data=fo)

sns.countplot(x='medal', data=fo, hue='Gender')



Indexing :-

1) type (fo.index) → RangeIndex

fo.index[100] = give some value

Index are immutable

2) set_Index (keys, inplace = True) → Replace on current value

0
1
2
3
4
} Index

fo.set_Index ('Athelte')

first col become Athelte

3) reset_index () → Reset the set index using inplace=False

→ df.reset_index (inplace = True)

sort_index()

df.sort_index(inplace=True)

→ replace and sort the index

loc[] → Dataframe.loc[]

loc[] will raise a KeyError when the items are not found

→ first set the index by the inplace

→ df.[df.Alethe = "USA2V Bolt"]

give some result as above

→ df.loc[df.Alethe = "US.7AV.002]"

iloc[] → gives specific result of list of index.

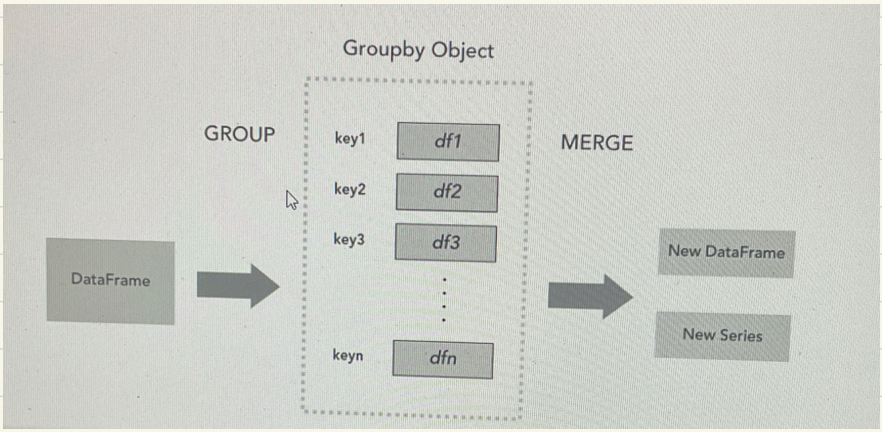
df.iloc[[152, 222, 1500)]

shows specific result of index given in list.

df.loc[1:4] → Show 1, 2, 3 index parts

Groupby → split Dataframe into groups
→ Apply function to each group independently.
→ Combine the result into Dataframe

Returns a object



df.groupby('Edition')

To view whats in group:-

list(df.groupby('Edition'))

Iterate through a group

```
for key, group in DataFrame.groupby():
    print(key)
    print(group)
```

```
for group_key, group_value in df.groupby('Edition'):
    print(group_key)
    print(group_value)
```

type = DataFrame

groupby

```
df.groupby('Edition').size()
```

```
df.groupby(["Edition", 'NOC', 'Medal']).
    agg('count')
```

df.loc[df.Athlete == 'LEWIS, Carl'].groupby('Athlete').agg({'Edition': ['min', 'max', 'count']})

#Reshaping:- stack / unstack

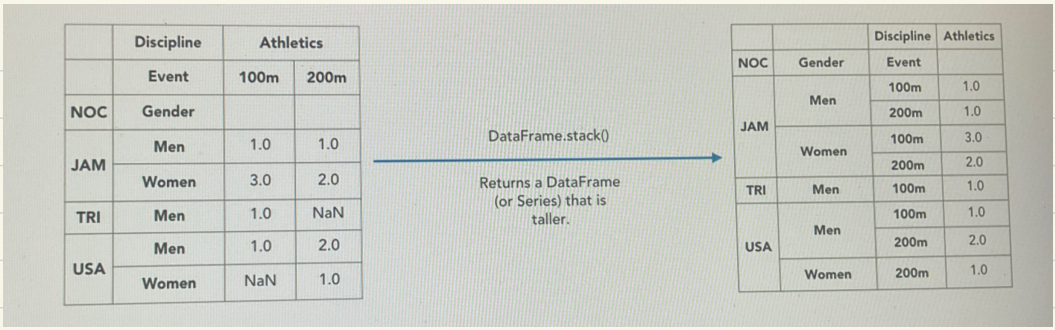
Stack – allow to move innermost col to the rows for the dataframe

-) Returns dataframe or series

-) by default dropna = True

no NA value will be show

-) Pivot a level of the column labels, returning a dataframe or series, with a new innermost level of row labels.



	Discipline	Athletics	
	Event	100m	200m
NOC	Gender		
JAM	Men	1.0	1.0
	Women	3.0	2.0
TRI	Men	1.0	NaN
USA	Men	1.0	2.0
	Women	NaN	1.0

		Discipline	Athletics
NOC	Gender	Event	
JAM	Men	100m	1.0
		200m	1.0
	Women	100m	3.0
		200m	2.0
TRI	Men	100m	1.0
USA	Men	100m	1.0
		200m	2.0
	Women	200m	1.0

df.stack()

Unstack :-

- return series and dataframe
- missing data values are displayed
- unstack the stack data

#challenges

```
import pandas as pd
df = pd.read_csv("")
print(df.head(10)
```