src: dpoint 4t channel

Data manipulation, Data Filtering

dataframe is a collection of series.

df.dtypes

col name | datatype

:

dtype:

df.describe()

↳ applied only on numerical columns

↓

it describes

mean, std, count, min, 25%, 50%, 75%, max

> df.dtypes == 'object'

Ex:

col1	False → not an object dtype
col2	True → object.dtype

> (df.dtypes == 'object').dtypes → # bool dtype ('bool')

> df.dtypes [object index]

↓

df.dtypes == 'object'

→

col5 object
col6 object
dtype: object

> df.dtypes [object index].index

only prints the values

→ Index (['col5', 'col6'], dtype = 'object')

Accessing columns:

df['col1'] # for accessing a single column

df[['col1','col2',...]] # for accessing 1 or more columns.

Accessing Rows:

df['col1'][0] → first value of col1

df[0] → first row

df[0:] → all rows

df[0:n]
↳ upper bound.
lower bound

→ ~~df['col']~~

df['newcol'] = 'dpoint'
↳ Assigned to every row in newcol.

→ pd.Categorical(df['col6'])

→ like levels in R lang

→ df['col6'].unique()

→ df['col6'].isnull().

Visualisation

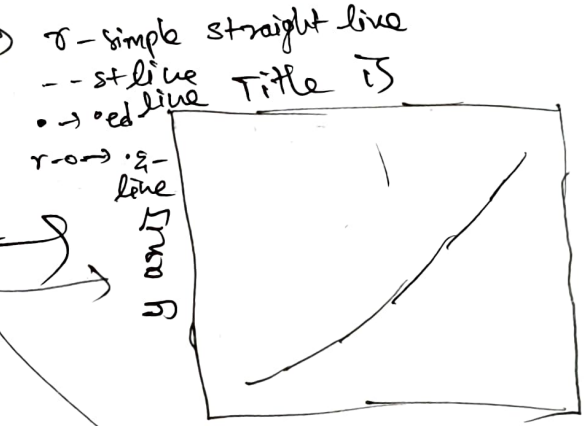
imports → numpy
pandas
seaborn
matplotlib.pyplot

[%matplotlib inline]
↳ To display inline plots for Jupyter notebooks

from pandas.plotting import scatter_matrix.

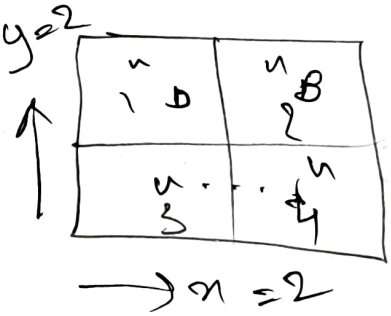
from numpy.random import randn, randint, uniform, sample

```
x = [1,2,3,4,5]
y = [a**2 for a in x] → set of squares of x.
    [1,4,9,16,25]
```



plt.subplot(_x, _y, _n)

(2,2,1) → D
2 → B



Object oriented drawing

fig = plt.figure()

↓

object.

axes = fig.add_axes([1,1,1,1]) or([0.5,0.5,0.5,0.5])

axes.plot(x,y,'b') (x,y,'g')

axes.set_xlabel('x axis') → blue

axes.set_ylabel('y axis') →

both these plots are on same canvas

in figure() → edgecolor = 'r' red border

linewidth → linewidth of edgecolor

→ if axes are not set properly, then plots may overlap.

df = pd.DataFrame(randn(1000), index=np.linspace(1,1000,1000), columns=['value'])

ts = pd.Series(randn(1000), index=np.linspace(1,1000,1000))

df.describe()