Introduction – Part 1

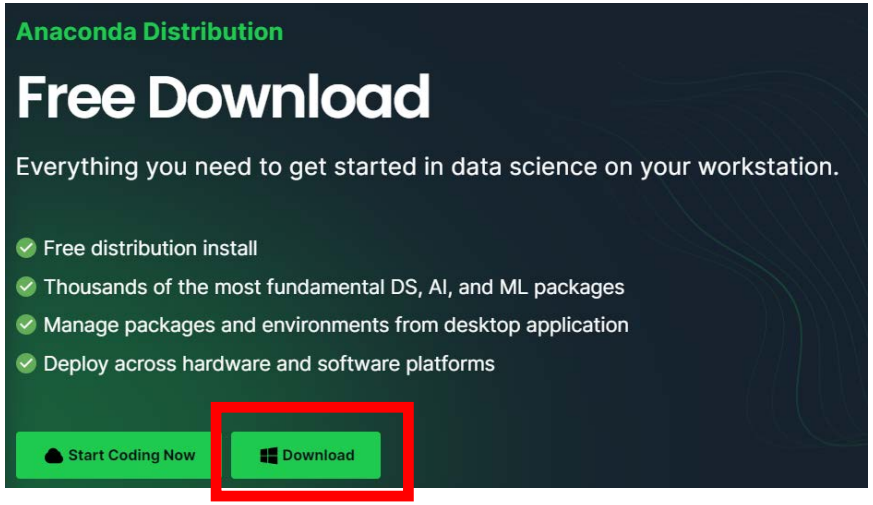
How to install and use Jupyter Notebook

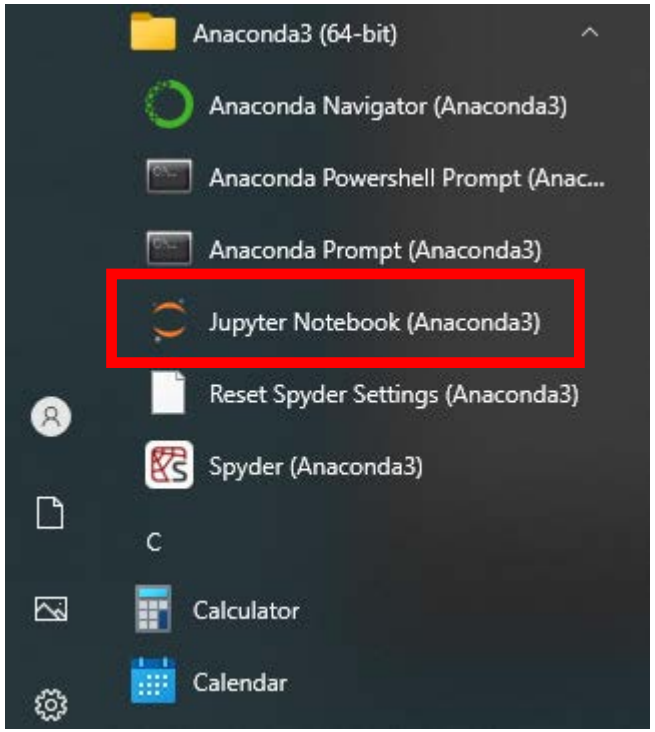
Go to https://www.anaconda.com/download

Download **Anaconda3**

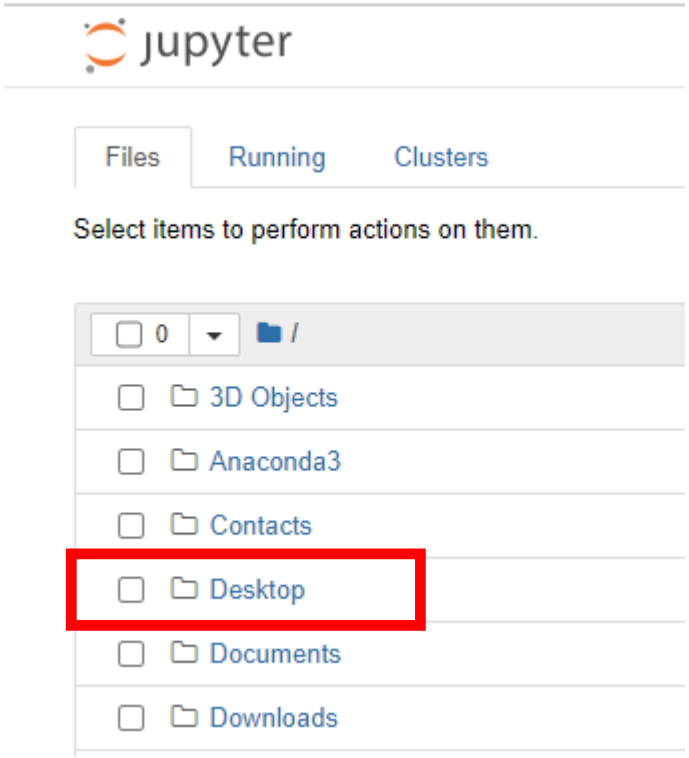
Install **Anaconda3**

Click **Jupyter Notebook**

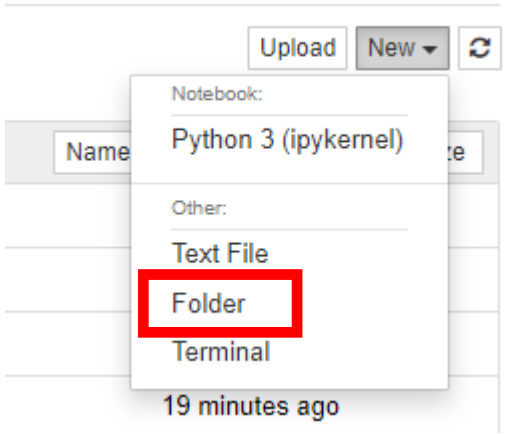




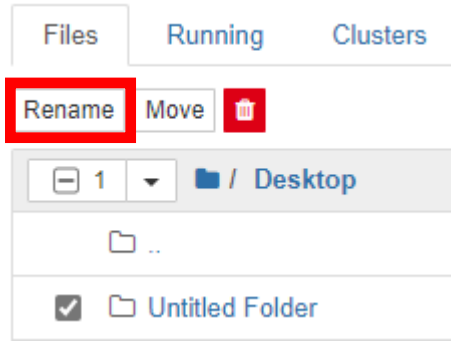
Click Desktop

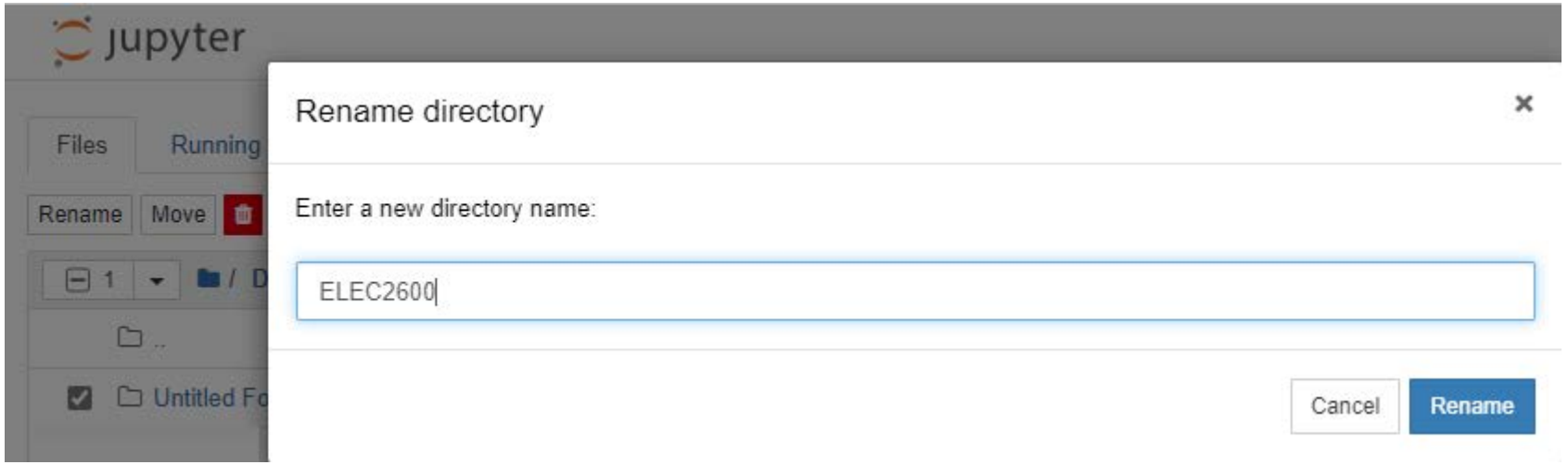


Create a Folder

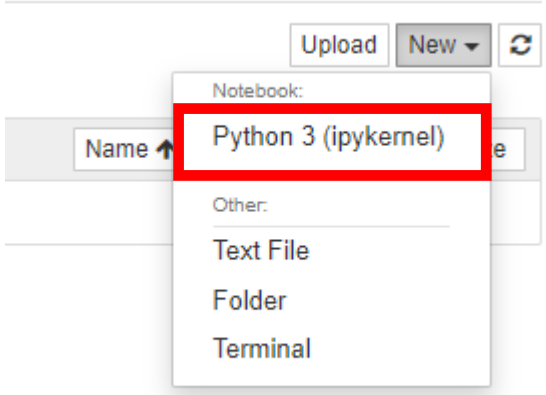


Rename the Folder

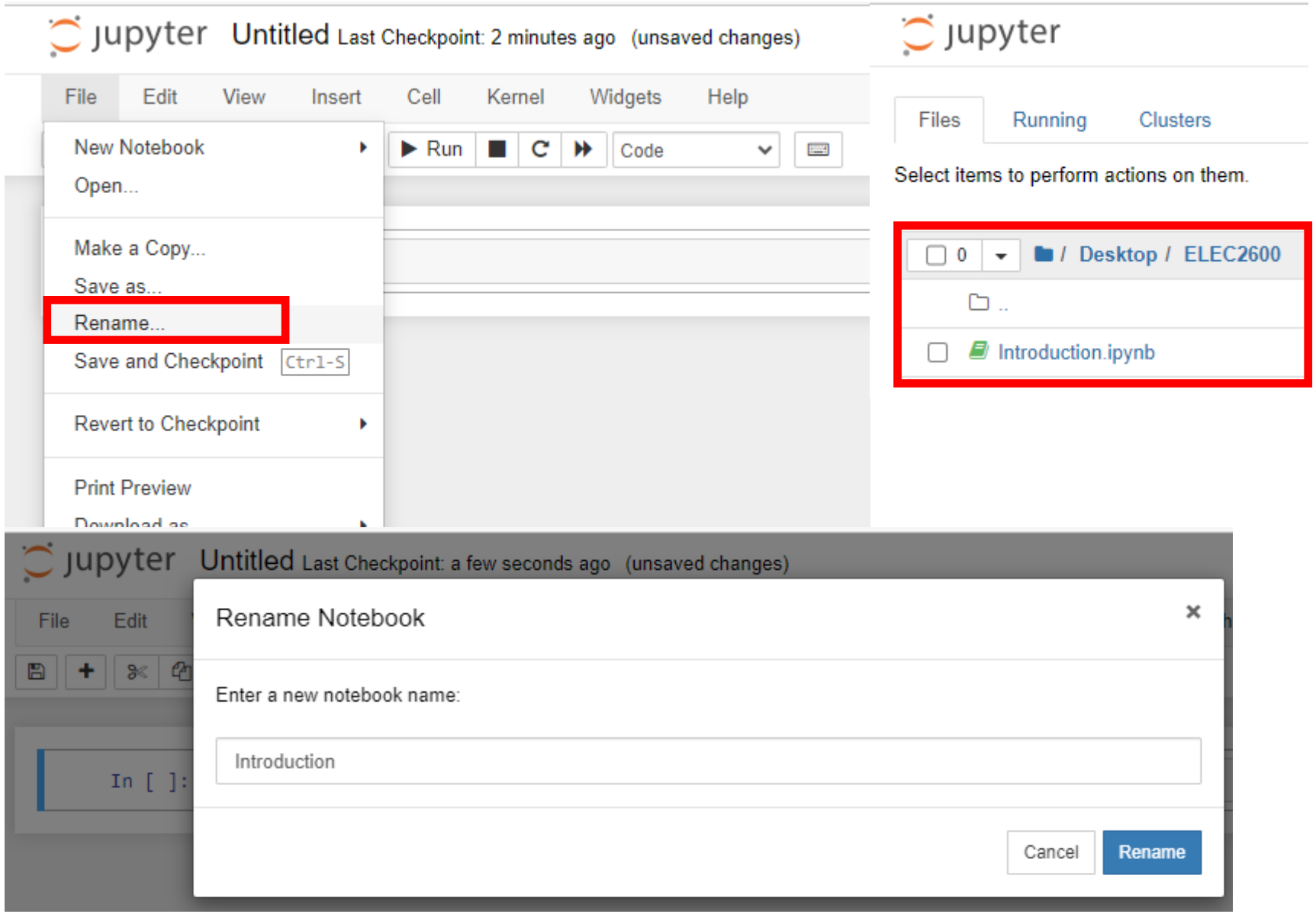




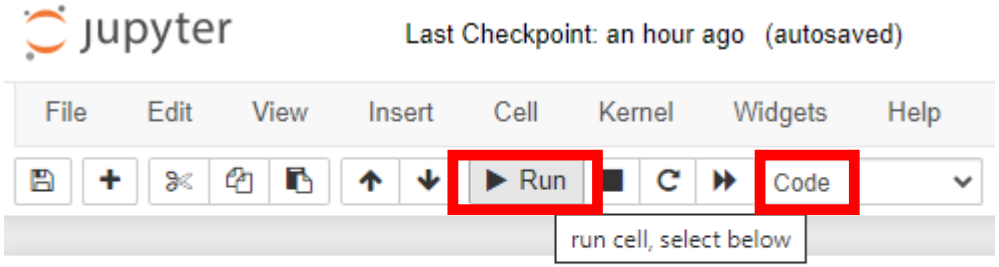
Create a new notebook

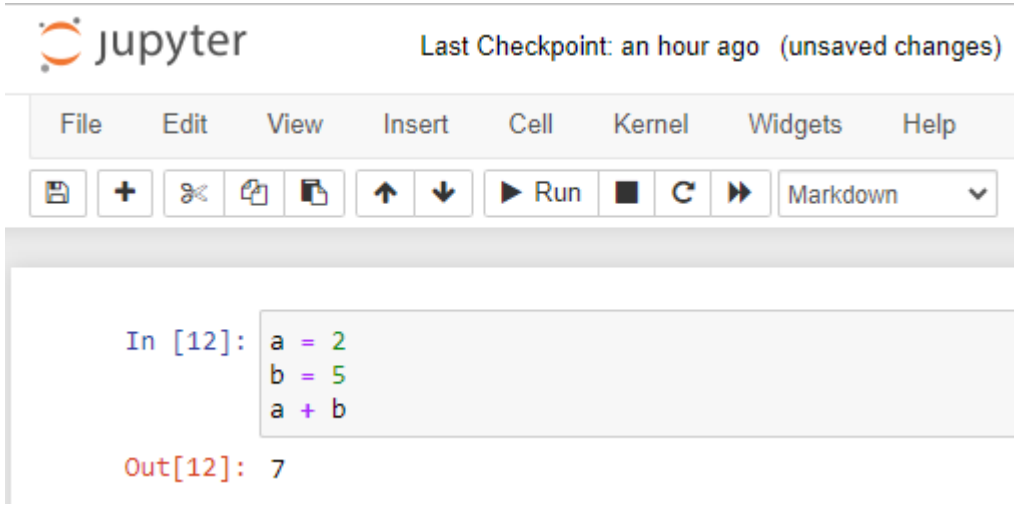


Rename the notebook

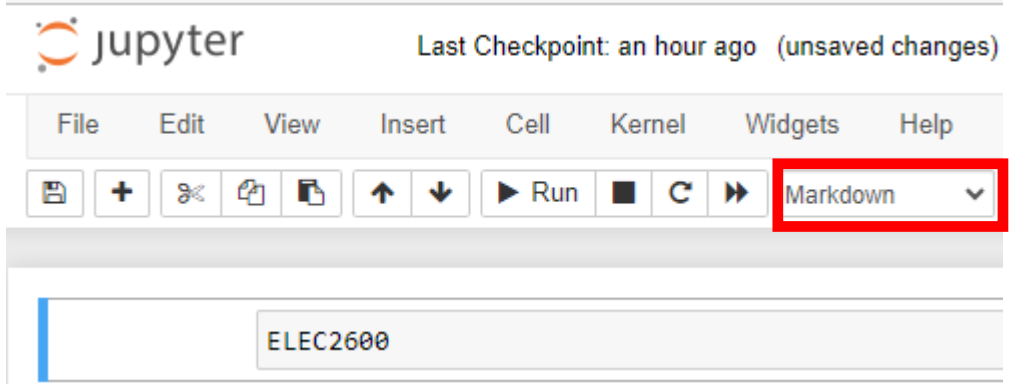


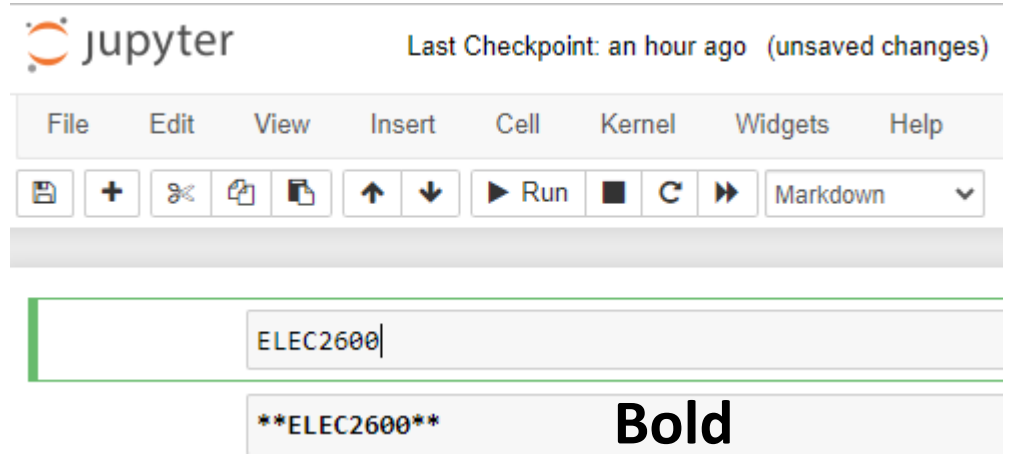
Program code and Run cell



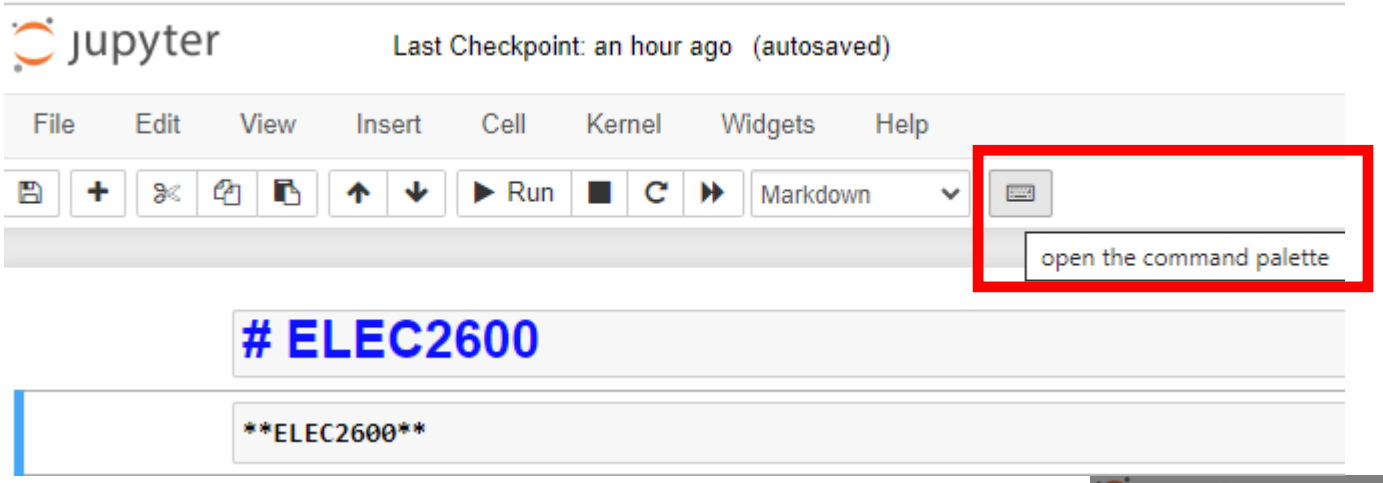


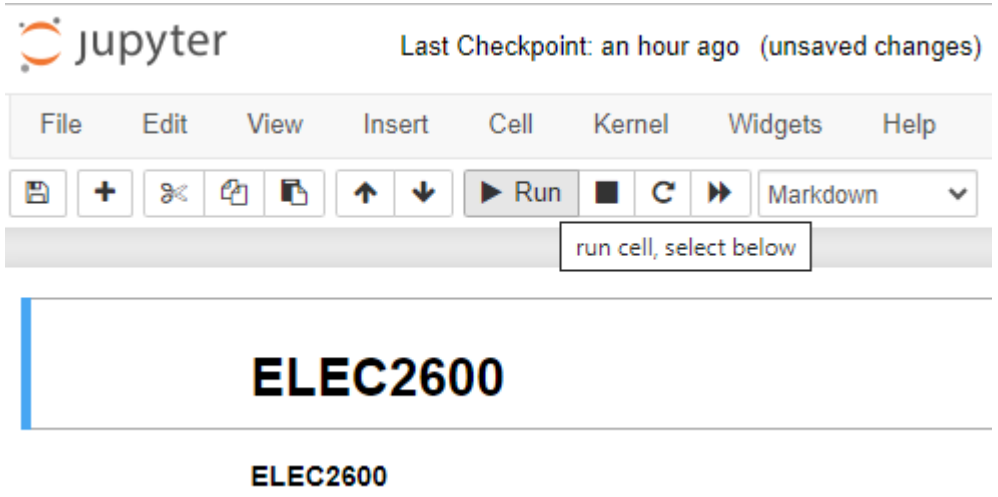
Text (Bold)

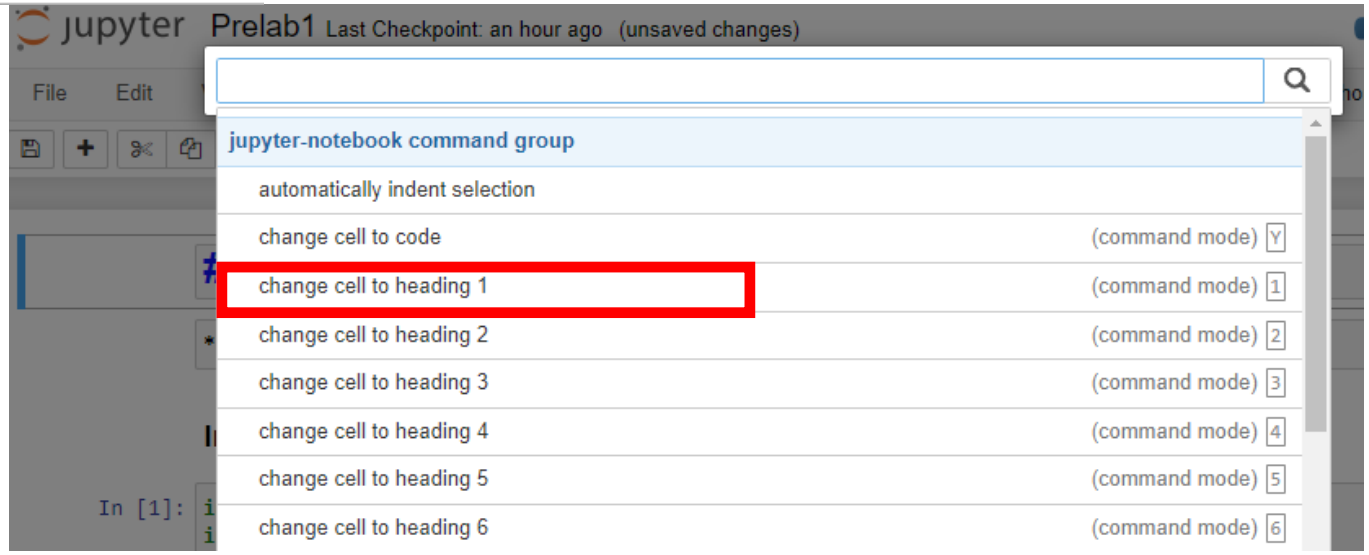




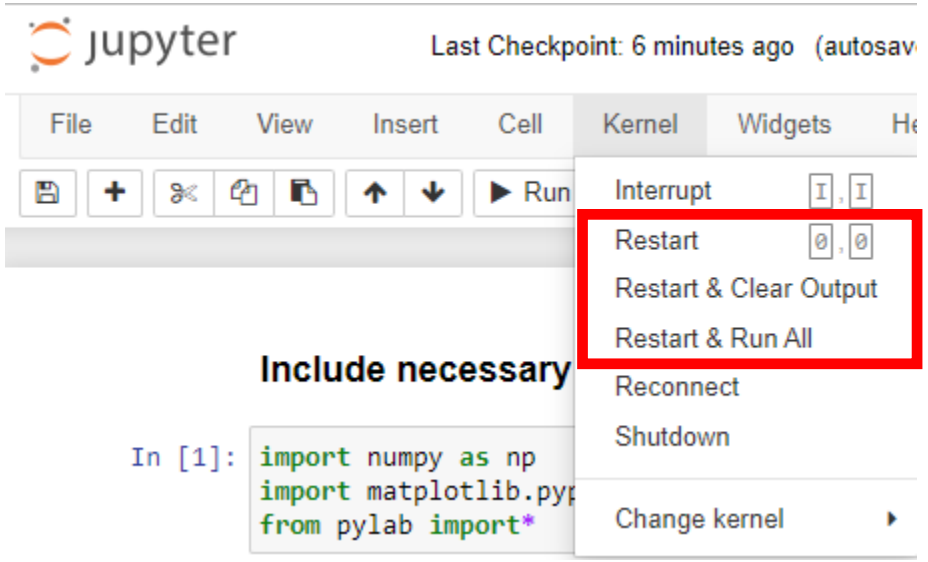
Text (Heading)

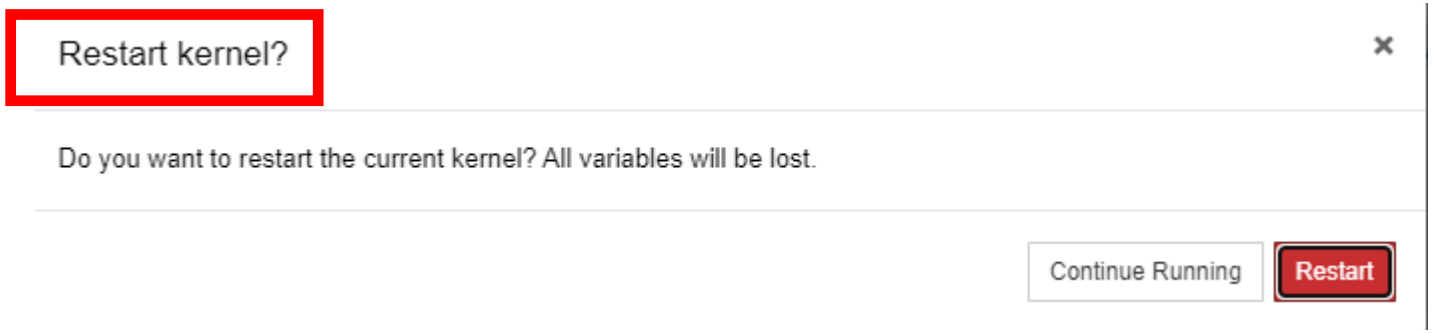


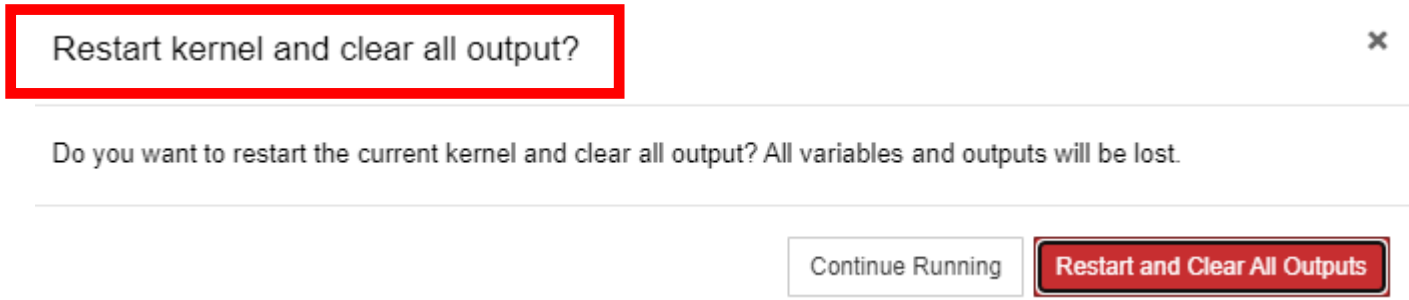


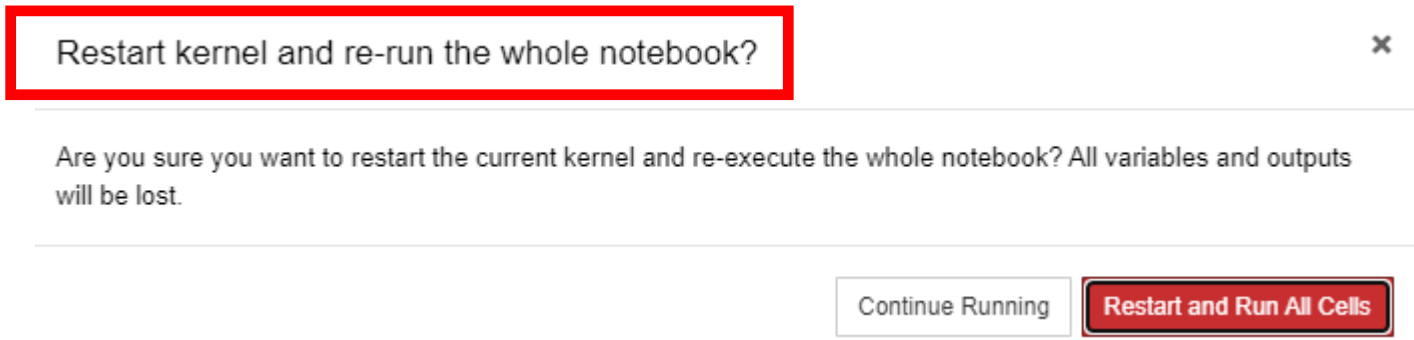


Kernel

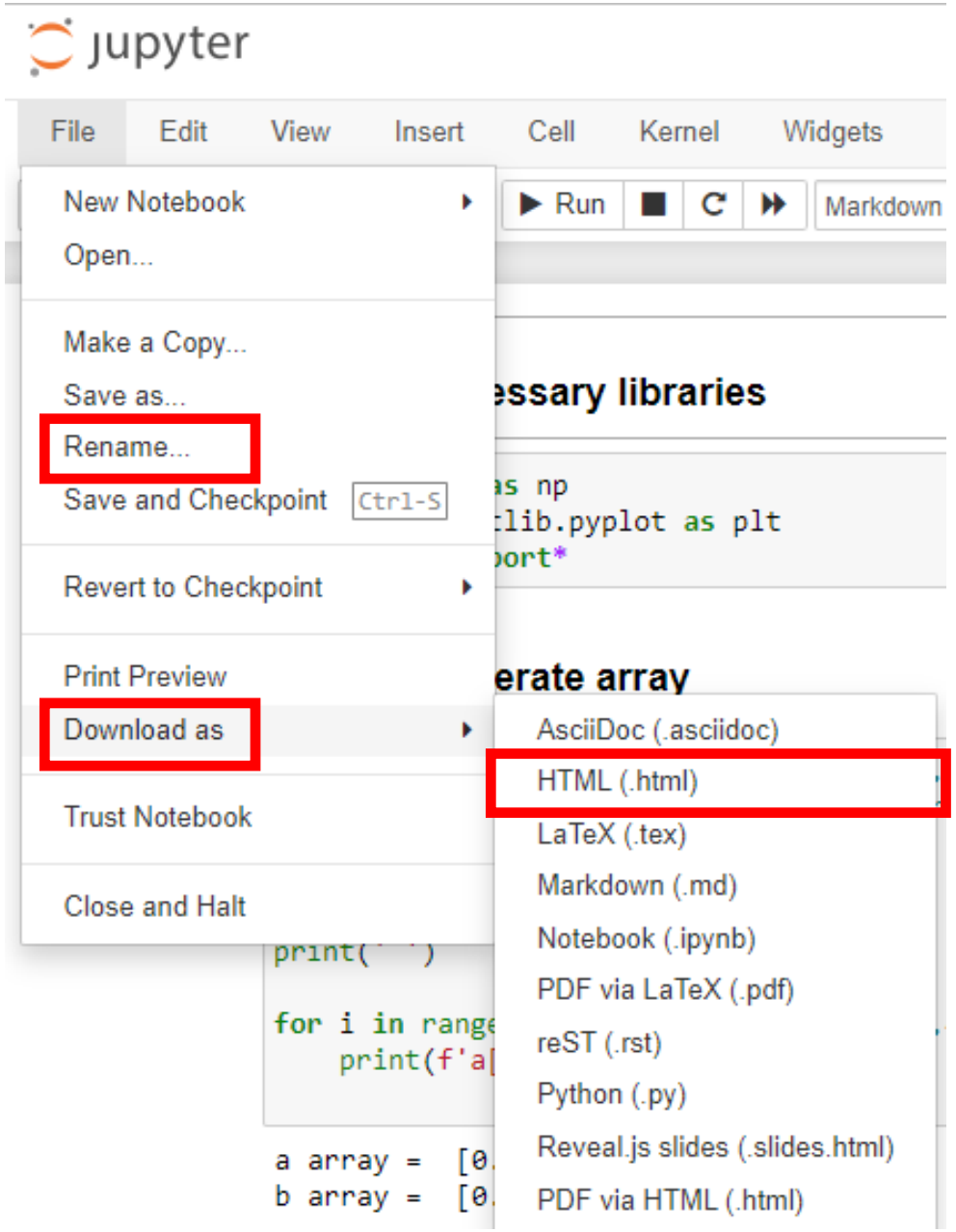

jupyter
Last Checkpoint: 6 minutes ago (autosav
File
Edit
View
Insert
Cell
Kernel
Widgets
Run
Interrupt
Restart
Restart & Clear Output
Restart & Run All
Reconnect
Shutdown
Change kernel
Include necessary
In [1]:
import numpy as np
import matplotlib.pyp
from pylab import*


Restart kernel?
Do you want to restart the current kernel? All variables will be lost.
Continue Running
Restart

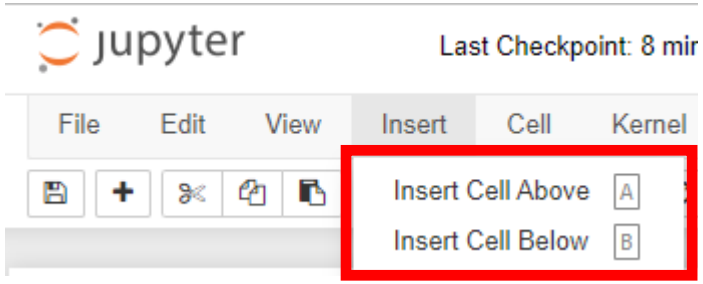

Restart kernel and clear all output?
Do you want to restart the current kernel and clear all output? All variables and outputs will be lost.
Continue Running
Restart and Clear All Outputs


Restart kernel and re-run the whole notebook?
Are you sure you want to restart the current kernel and re-execute the whole notebook? All variables and outputs will be lost.
Continue Running
Restart and Run All Cells

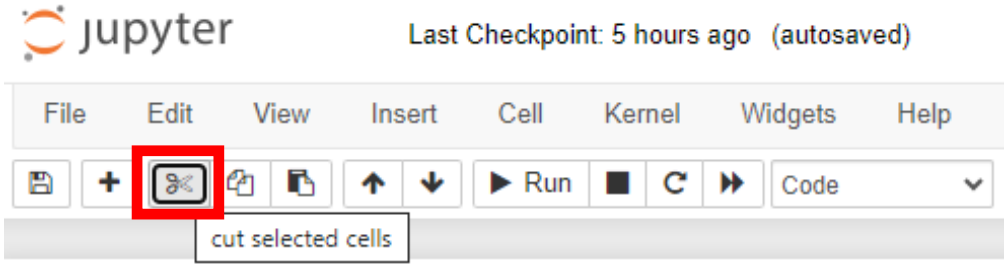
File



Insert



Delete a cell



Ex. 06

```
a = np.array([1, 4, 5])
b = np.array([1, 4, 5, 5, 4, 1, 1, 1])

for i in range(a.size):
    print(f'a{[i]} = ', a[i])
    compare = np.equal(b, a[i])
    print(compare)
    count = np.cumsum(compare)
    print(count)
    print(' ')
```

np.equal

np.cumsum

```
a[0] =  1
[ True False False False False  True  True  True]
[1 1 1 1 1 2 3 4]

a[1] =  4
[False  True False False  True False False False]
[0 1 1 1 2 2 2 2]

a[2] =  5
[False False  True  True False False False False]
[0 0 1 2 2 2 2 2]
```

How to obtain the relative frequency versus *n* ?

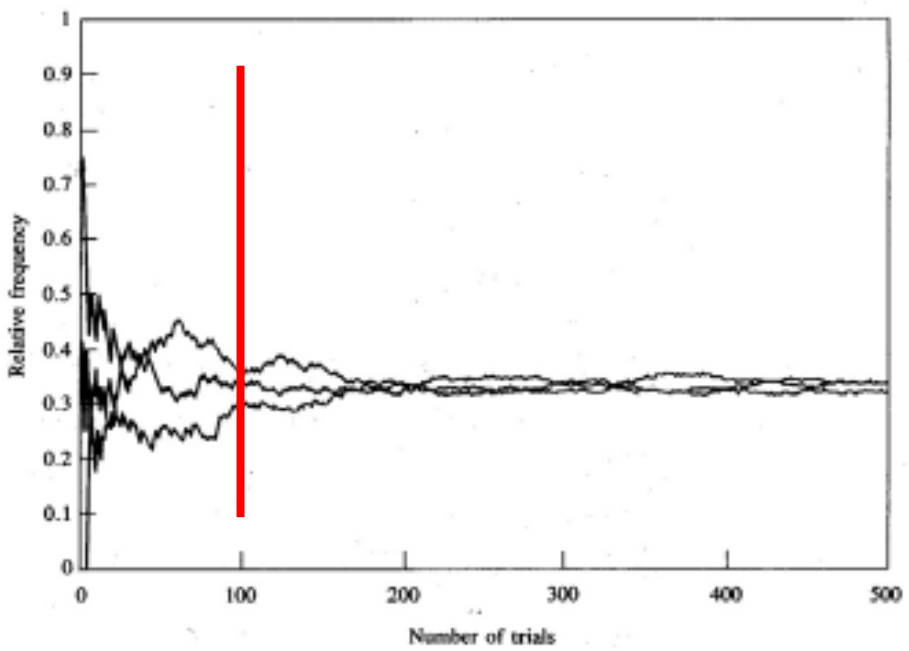
How to plot the relative frequency versus *n* ?

Relative frequency

- Let $N_1(n)$, $N_2(n)$ and $N_3(n)$ be the *number of times* that we pick balls 1, 2, and 3 in n trials (***events***).
- Define the *relative frequency* of the outcome k as $f_k(n) = \frac{N_k(n)}{n}$
- This experiment exhibits *statistical regularity:* as n increases, the relative frequency approaches a ***constant value***

$$\lim_{n\to\infty} f_k(n) = p_k$$

where p_k indicates the probability of outcome k.



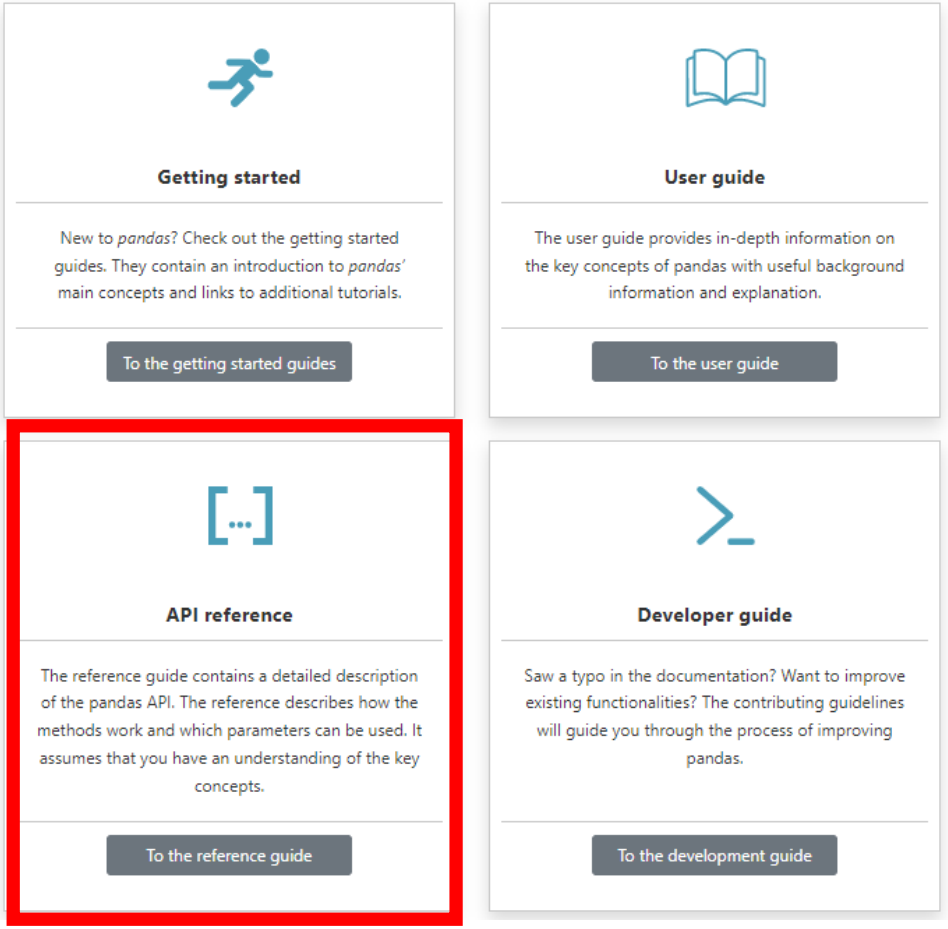
Provides a key connection between measurement of physical quantities and probability models!

Introduction – Part 2

How to handle excel file

Go to https://pandas.pydata.org/pandas-docs/stable/index.html

Click **API reference**



Click **DateFrame**

Click and read the description of the followings (groupby, count and sum)

Function application, GroupBy & window

`DataFrame.apply` (func[, axis, raw, ...])	Apply a function along an axis of the DataFrame.
`DataFrame.applymap` (func[, na_action])	Apply a function to a Dataframe elementwise.
`DataFrame.pipe` (func, *args, **kwargs)	Apply chainable functions that expect Series or DataFrames.
`DataFrame.agg` ([func, axis])	Aggregate using one or more operations over the specified axis.
`DataFrame.aggregate` ([func, axis])	Aggregate using one or more operations over the specified axis.
`DataFrame.transform` (func[, axis])	Call `func` on self producing a DataFrame with the same axis shape as self.
`DataFrame.groupby` ([by, axis, level, ...])	Group DataFrame using a mapper or by a Series of columns.
`DataFrame.rolling` (window[, min_periods, ...])	Provide rolling window calculations.
`DataFrame.expanding` ([min_periods, axis, method])	Provide expanding window calculations.
`DataFrame.ewm` ([com, span, halflife, alpha, ...])	Provide exponentially weighted (EW) calculations.

Computations / descriptive stats

`DataFrame.abs` ()	Return a Series/DataFrame with absolute numeric value of each element.
`DataFrame.all` ([axis, bool_only, skipna])	Return whether all elements are True, potentially over an axis.
`DataFrame.any` (*[, axis, bool_only, skipna])	Return whether any element is True, potentially over an axis.
`DataFrame.clip` ([lower, upper, axis, inplace])	Trim values at input threshold(s).
`DataFrame.corr` ([method, min_periods, ...])	Compute pairwise correlation of columns, excluding NA/null values.
`DataFrame.corrwith` (other[, axis, drop, ...])	Compute pairwise correlation.
`DataFrame.count` ([axis, numeric_only])	Count non-NA cells for each column or row.
`DataFrame.sem` ([axis, skipna, ddof, numeric_only])	Return unbiased standard error of the mean over requested axis.
`DataFrame.skew` ([axis, skipna, numeric_only])	Return unbiased skew over requested axis.
`DataFrame.sum` ([axis, skipna, numeric_only, ...])	Return the sum of the values over the requested axis.
`DataFrame.std` ([axis, skipna, ddof, numeric_only])	Return sample standard deviation over requested axis.

Ex.01 – Groupby and Count

```
print(data)
```

```
            Class  Price
Index
1           fruit      5
2           fruit      5
3           fruit     10
4           fruit     10
5           fruit      3
6       vegetable      1
7       vegetable      1
8       vegetable      5
9       vegetable      2
10      vegetable      4
11          fruit      5
12          fruit      5
13      vegetable      1
14      vegetable      5
15      vegetable      2
16      vegetable      4
```

Ex.01 - Groupby and Count

```
n_class = data.groupby('Class')['Price'].count()
print(n_class)
print(' ')

print('Number of fruits = ', n_class['fruit'])
print(' ')

print('Total number of products', sum(n_class))
```

```
Class
fruit        7
vegetable    9
Name: Price, dtype: int64

Number of fruits =  7

Total number of products 16
```

Relative frequency ?

Ex.02 – Groupby and Sum

```
print(data)
print(' ')

n_class = data.groupby('Class')['Price'].sum()
print(n_class)
```

Total price of fruit = ?

Total price of vegetable = ?

Index	Class	Price
1	fruit	5
2	fruit	5
3	fruit	10
4	fruit	10
5	fruit	3
6	vegetable	1
7	vegetable	1
8	vegetable	5
9	vegetable	2
10	vegetable	4
11	fruit	5
12	fruit	5
13	vegetable	1
14	vegetable	5
15	vegetable	2
16	vegetable	4

```
Class
fruit        43
vegetable    25
Name: Price, dtype: int64
```

Ex.03 – Create a new column

```
data['data_price'] = data['Price'] > 4
print(data)
print(' ')

print('The sum of data_price = ', sum(data['data_price']))
print(' ')

n_fruit_4 = data.groupby('Class')['data_price'].sum()
print(n_fruit_4)
```

True = 1 **False = 0**

fruit (Price > 4) + vegetable (Price > 4) = 8

fruit (Price > 4) = ?

vegetable (Price > 4) = ?

Index	Class	Price	data_price
1	fruit	5	True
2	fruit	5	True
3	fruit	10	True
4	fruit	10	True
5	fruit	3	False
6	vegetable	1	False
7	vegetable	1	False
8	vegetable	5	True
9	vegetable	2	False
10	vegetable	4	False
11	fruit	5	True
12	fruit	5	True
13	vegetable	1	False
14	vegetable	5	True
15	vegetable	2	False
16	vegetable	4	False

```
The sum of data_price =  8
```

```
Class
fruit        6
vegetable    2
Name: data_price, dtype: int64
```

Event A **Event B**

Prior probability of event A $P(A)$

Prior probability of event B $P(B)$

Event A and Event B both occur $P(A \cap B)$

Conditional probability $$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

What is the prior probability of fruit ?

What is the probability $P[(\text{Price} > 4) \cap \text{Fruit}]$?

What is the conditional probability $P[(\text{Price} > 4) \mid \text{Fruit}]$?