

**Understanding and using code notebooks**

# Extensive data science with Python scripts become challenging quickly

1 Comments in your script tend to be terse or non-existent

2 Intermediate results are not immediately accessible

3 Coworkers spend a lot of time reading your comments

4 You struggle to find the right plot in your output folder, because there are so many

```
LE6_Aggregate v2.py
import datetime
myData["TransDate"] = myData["TransDate"].apply(datetime.datetime.strptime, dayfirst=True)

#####
Part 1: Aggregate on datasets
#####
myData.loc[myData["Customer"]==149236,]
#myData.loc[myData["Customer"]==149236,].groupby("Customer")["PurchAmount"].sum()

### Slide 10: Sum all purchase amounts by customer
myData.groupby("Customer", as_index=False)["PurchAmount"].sum()

### Slide 11: Aggregate a variable by one dimension
myData.groupby("Customer", as_index=False)["PurchAmount"].agg(["sum"]).rename(columns={"PurchAmount": "sum"})

### Slide 12: Apply multiple aggregation functions to a variable by one dimension
myData.groupby("Customer", as_index=False)["PurchAmount"].agg({"AggPurch": "sum", "PurchAmount": "max"})

### Slide 13: Apply an aggregating function to the whole dataset
myData["PurchAmount"].sum()

### Slide 15: Create new columns with the transform()-function
myData["AggPurch"] = myData.groupby("Customer")["PurchAmount"].transform(sum)
myData

### This will create NAs
myData["AggPurch_NAs"] = myData.groupby("Customer")["PurchAmount"].sum()
myData["AggPurch_NAs"]

### Slide 16: Apply an aggregating function to multiple variables
myData.groupby("Customer")["PurchAmount", "Quantity"].sum()

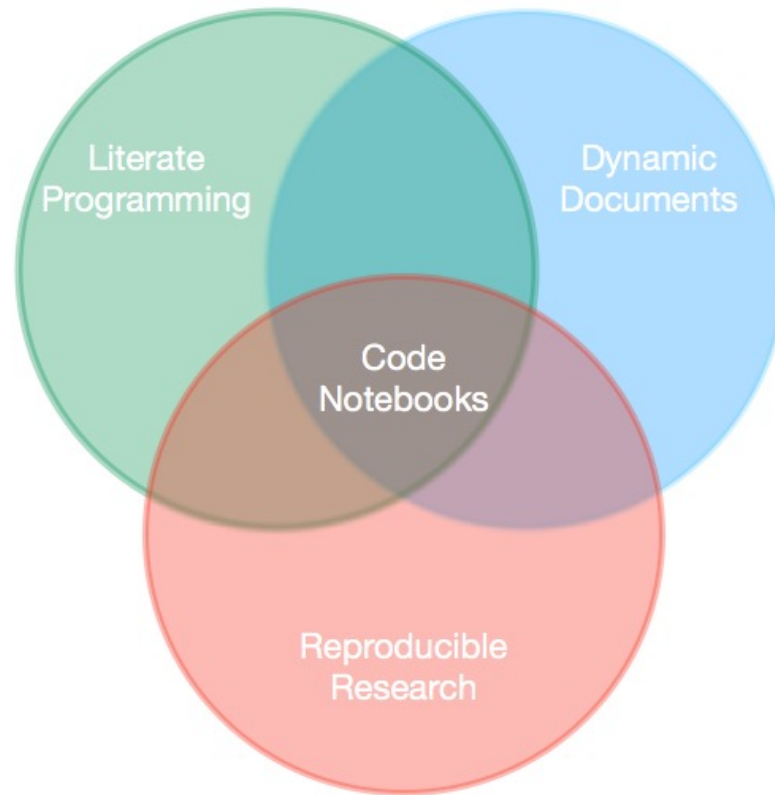
### Slide 16: Apply an aggregating function to multiple variables
myData.groupby("Customer", as_index=False).agg({"PurchAmount": "max", "sum"}, "Quantity": "sum")

### Slide 17: Aggregate multiple variables by two dimensions
myData.groupby(["Customer", "TransDate"])["PurchAmount", "Quantity"].sum()

### Slide 18: Aggregation for subset of rows
myData.iloc[0:5].groupby("Customer", as_index=False)["PurchAmount"].sum()
```

Scripts for data science can become messy quickly.

**Code notebooks are a tool to report your code and results and make them publicly available**



# Code notebooks help you to:

- Divide your code into manageable chunks.
  - Execute small chunks of code one by one.
  - Evaluate and store preliminary results.
- Take notes along the process in rich text format (paragraphs, equations, tables, ... ).
- Present and discuss your results and code chunks with colleagues.



# What do you get out of code notebooks?

## Example: Jupyter Notebook

This is the original look of a Jupyter Notebook- Google Colab uses Jupyter Notebooks in a slightly re-designed way (but the technology under the hood is the same)

1  
Exporting your document will produce immediate results of your calculations

2  
Different export formats, e.g. HTML or PDF

### Exercise Part I

#### Orders in 2015

Only certain business segments were considered:

- Food
- Beverages
- Hygiene

Other business segments were **not** considered:

- Clothes
- Accessories

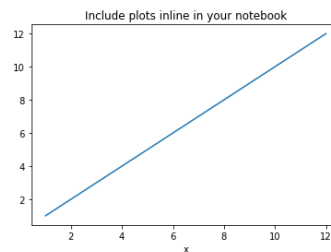
#### Include Plots

```
In [4]: import matplotlib.pyplot as plt
import matplotlib inline
x=[1,3,4,5,7,10,12]
plt.plot(x,x)
plt.title("Include plots inline in your notebook")
plt.xlabel("x")
plt.show()
```

3  
Easy text formatting with Markdown

4  
Integration of Python code and other languages, e.g. SQL and R

5  
Output of code chunks is directly displayed



6  
Put your work in a nice format

# Notebooks versus regular programming scripts

Activity	Scripts	Notebooks
Building a narrative	<b>Comments</b> are available to describe code.	<b>Rich text</b> is available to describe the process and conclusion
Manage output	Output is available in the console or environment.	<b>Output is embedded</b> in a single document. Code and output may be divided into separate <b>code chunks</b> .
Creating a final report	<b>Creating a report is a separate</b> and time-consuming <b>step</b> .	<b>Instant report possible.</b> Reports are publishable as HTML, PDF, ...)

Allows very descriptive and reproducible documentation <sup>1</sup>

Immediate update of output and report <sup>2</sup>

Easy sharing possible <sup>3</sup>

# Notebooks versus regular programming scripts

Activity	Scripts	Notebooks	
Building a narrative	<b>Comments</b> are available to describe code.	<b>Rich text</b> is available to describe the process and conclusion	Allows very descriptive and reproducible documentation <sup>1</sup>
Manage output	Output is available in the console or environment.	<b>Output is embedded</b> in a single document. Code and output may be divided into separate <b>code chunks</b> .	Immediate update of output and report <sup>2</sup>
Creating a final report	<b>Creating a report is a separate</b> and time-consuming <b>step</b> .	<b>Instant report possible.</b> Reports are publishable as HTML, PDF, ...)	Easy sharing possible <sup>3</sup>

# Notebooks versus regular programming scripts

Activity	Scripts	Notebooks
Building a narrative	<b>Comments</b> are available to describe code.	<b>Rich text</b> is available to describe the process and conclusion
Manage output	Output is available in the console or environment.	<b>Output is embedded</b> in a single document. Code and output may be divided into separate <b>code chunks</b> .
Creating a final report	<b>Creating a report is a separate</b> and time-consuming <b>step</b> .	<b>Instant report possible.</b> Reports are publishable as HTML, PDF, ...)

Allows very descriptive and reproducible documentation <sup>1</sup>

Immediate update of output and report <sup>2</sup>

Easy sharing possible <sup>3</sup>



# Notebooks versus regular programming scripts

Activity	Scripts	Notebooks
Building a narrative	<b>Comments</b> are available to describe code.	<b>Rich text</b> is available to describe the process and conclusion
Manage output	Output is available in the console or environment.	<b>Output is embedded</b> in a single document. Code and output may be divided into separate <b>code chunks</b> .
Creating a final report	Creating a report is a <b>separate</b> and time-consuming <b>step</b> .	<b>Instant report possible.</b> Reports are publishable as HTML, PDF, ...)

Allows very descriptive and reproducible documentation <sup>1</sup>

Immediate update of output and report <sup>2</sup>

Easy sharing possible <sup>3</sup>

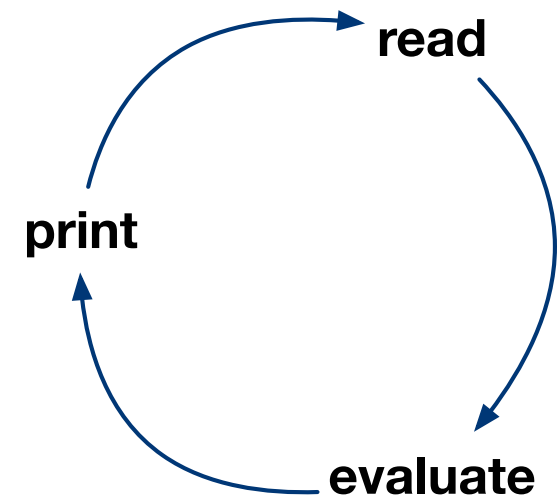
# Notebooks are based on the read-eval-print loop (REPL) technique

The notebook user-interface is developed on the **concept of interactive computing**:

1. User input
2. Input is evaluated
3. Result is printed and notebook is ready for next input

There are **two main features**:

- Store the last output for reuse
- Ability to save input and later reload and re-evaluate



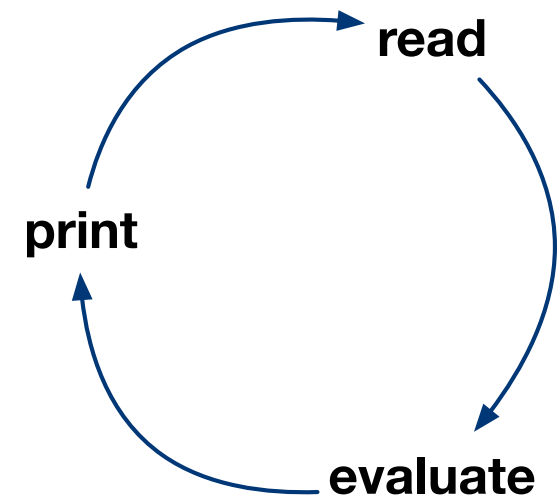
# Notebooks are based on the read-eval-print loop (REPL) technique

The notebook user-interface is developed on the **concept of interactive computing**:

1. User input
2. Input is evaluated
3. Result is printed and notebook is ready for next input

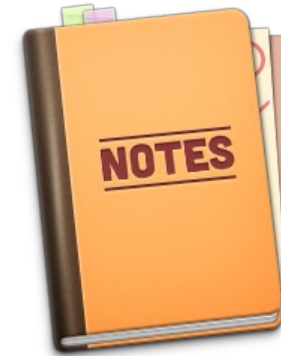
There are **two main features**:

- Store the last output for reuse
- Ability to save input and later reload and re-evaluate



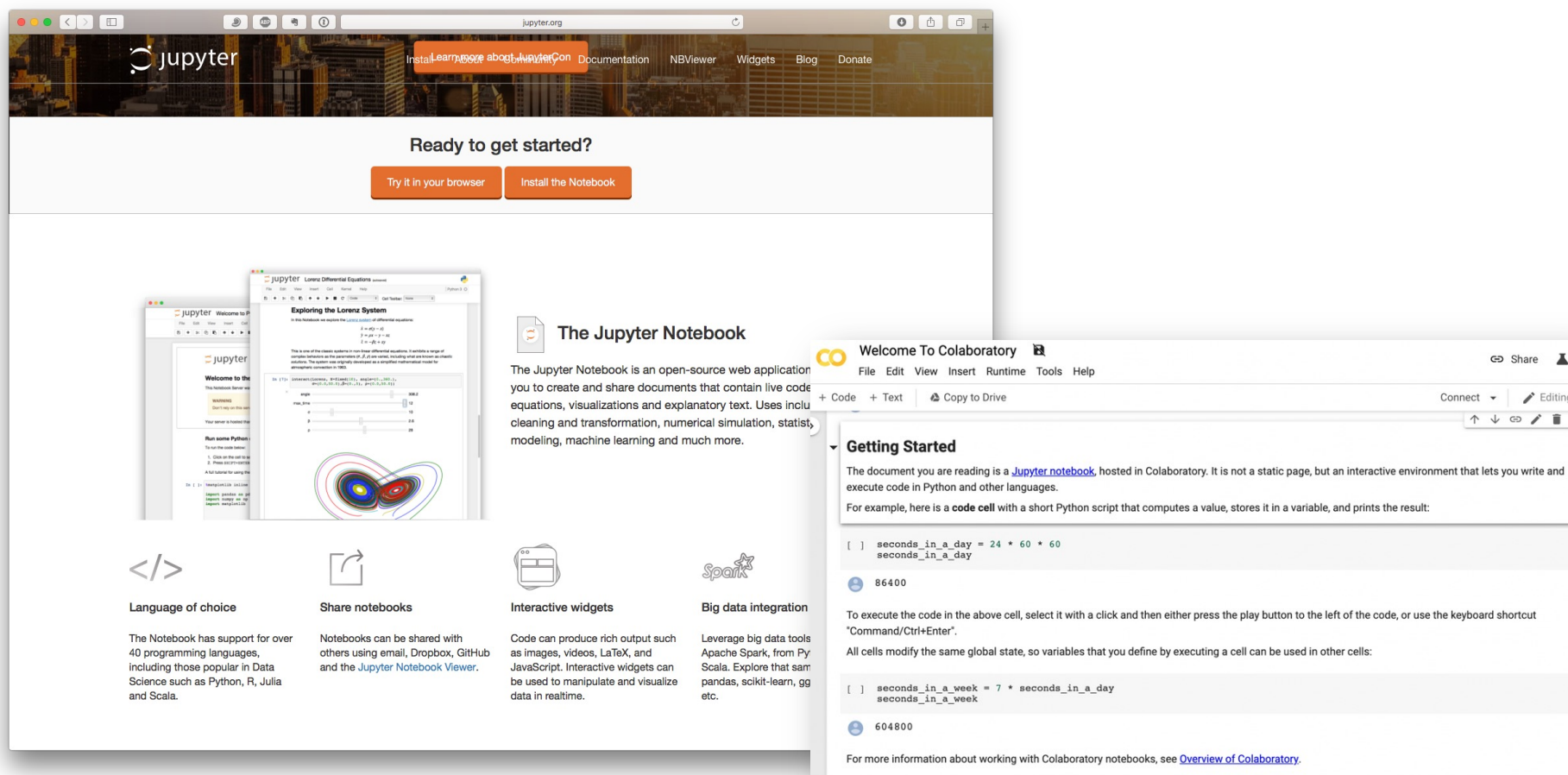
# There are many different notebooks, which mostly support multiple programming languages

- Jupyter (e.g. used by Google Colab; R, Python; around 40 different languages)
- R Notebooks (R, Python; around 8 different languages)
- Beaker (R, Python; around 17 different languages)
- Zeppelin (R, Python; around 20 different languages)



- Which one to pick depends on the requirements of the programming project and your personal preference. We pick to present Jupyter notebooks due to its integration into Spyder.

# For Python, Jupyter Notebooks (which are also used by service “Google Colab”) are the most popular



The image is a collage of three screenshots related to Jupyter Notebooks. The top-left screenshot shows the Jupyter.org homepage with the text "Ready to get started?" and buttons for "Try it in your browser" and "Install the Notebook". The bottom-left screenshot shows a Jupyter Notebook interface titled "Exploring the Lorenz System" with a plot of the Lorenz attractor. The right screenshot shows a Google Colaboratory interface titled "Welcome To Colaboratory" with a "Getting Started" section.

**The Jupyter Notebook**

The Jupyter Notebook is an open-source web application that lets you create and share documents that contain live code, equations, visualizations and explanatory text. Uses include data cleaning and transformation, numerical simulation, statistical modeling, machine learning and much more.

**Language of choice**

The Notebook has support for over 40 programming languages, including those popular in Data Science such as Python, R, Julia and Scala.

**Share notebooks**

Notebooks can be shared with others using email, Dropbox, GitHub and the [Jupyter Notebook Viewer](#).

**Interactive widgets**

Code can produce rich output such as images, videos, LaTeX, and JavaScript. Interactive widgets can be used to manipulate and visualize data in realtime.

**Big data integration**

Leverage big data tools Apache Spark, from PySpark. Explore that same pandas, scikit-learn, gg etc.

**Welcome To Colaboratory**

File Edit View Insert Runtime Tools Help

+ Code + Text Copy to Drive

Connect Editing

**Getting Started**

The document you are reading is a [Jupyter notebook](#), hosted in Colaboratory. It is not a static page, but an interactive environment that lets you write and execute code in Python and other languages.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
```

86400

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter".

All cells modify the same global state, so variables that you define by executing a cell can be used in other cells:

```
[ ] seconds_in_a_week = 7 * seconds_in_a_day
seconds_in_a_week
```

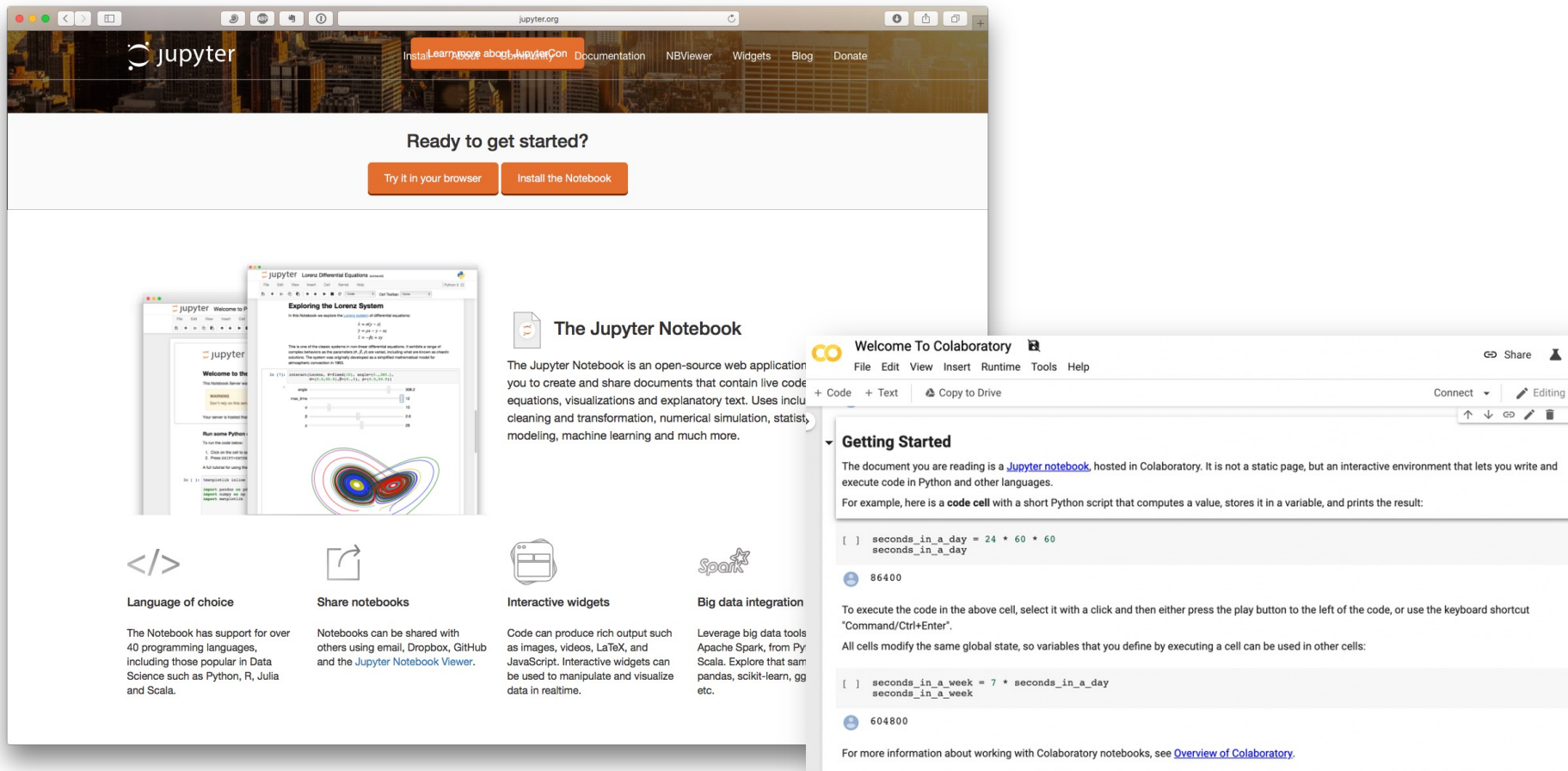
604800

For more information about working with Colaboratory notebooks, see [Overview of Colaboratory](#).

<http://jupyter.org>

<https://colab.research.google.com/>

# For Python, Jupyter Notebooks (which are also used by service “Google Colab”) are the most popular



The collage features three main components: the Jupyter.org homepage, a Jupyter Notebook interface, and a Google Colaboratory interface.

**Jupyter.org Homepage:** The top section has a navigation bar with links: [Install](#), [Learn more about JupyterCon](#), [Documentation](#), [NBViewer](#), [Widgets](#), [Blog](#), and [Donate](#). Below the navigation bar is a large banner with the text "Ready to get started?" and two buttons: "Try it in your browser" and "Install the Notebook".

**Jupyter Notebook Interface:** The interface shows a "Welcome to the Jupyter Notebook" message and a "Getting Started" section. It includes a code cell with a Python script that computes the value of `seconds_in_a_day` and prints the result. The output of the code is displayed below the cell.

**Google Colaboratory Interface:** The interface shows a "Welcome To Colaboratory" message and a "Getting Started" section. It includes a code cell with a Python script that computes the value of `seconds_in_a_day` and prints the result. The output of the code is displayed below the cell.

**Language of choice**  
The Notebook has support for over 40 programming languages, including those popular in Data Science such as Python, R, Julia and Scala.

**Share notebooks**  
Notebooks can be shared with others using email, Dropbox, GitHub and the [Jupyter Notebook Viewer](#).

**Interactive widgets**  
Code can produce rich output such as images, videos, LaTeX, and JavaScript. Interactive widgets can be used to manipulate and visualize data in realtime.

**Big data integration**  
Leverage big data tools Apache Spark, from PySpark. Explore that same pandas, scikit-learn, gg etc.

**Getting Started**  
The document you are reading is a [Jupyter notebook](#), hosted in Colaboratory. It is not a static page, but an interactive environment that lets you write and execute code in Python and other languages.  
For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
```

86400

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter".  
All cells modify the same global state, so variables that you define by executing a cell can be used in other cells:

```
[ ] seconds_in_a_week = 7 * seconds_in_a_day
seconds_in_a_week
```

604800

For more information about working with Colaboratory notebooks, see [Overview of Colaboratory](#).

<http://jupyter.org>

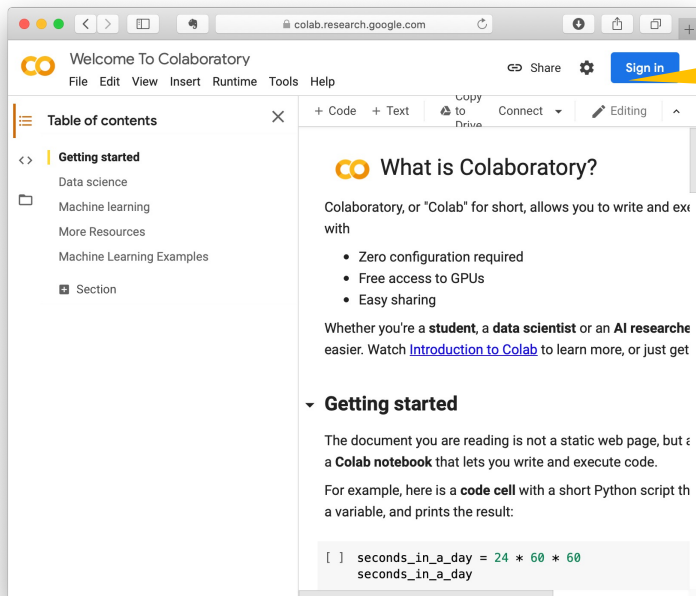
<https://colab.research.google.com/>

# Create your first Jupyter Notebook

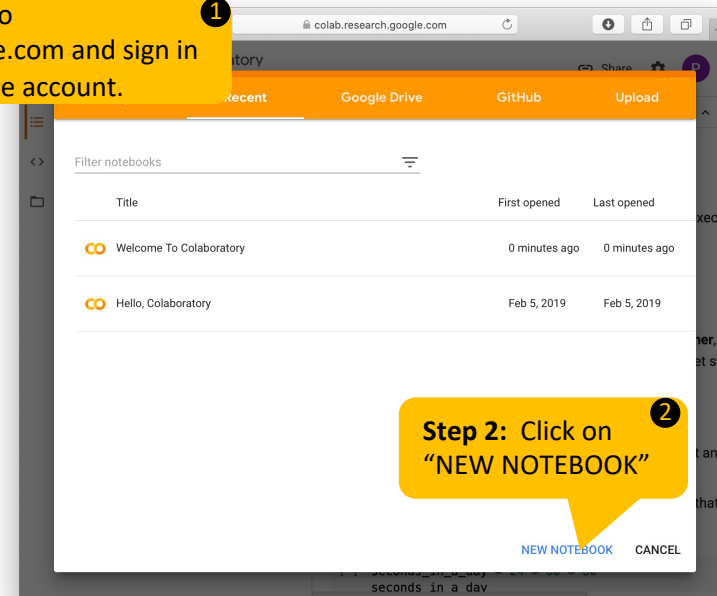
## Steps

1. **Create a new Jupyter Notebook**
2. Create Content
  - Add text elements as Markdown syntax
  - Add code elements in any supported program language
  - Use LaTeX in your Jupyter Notebook

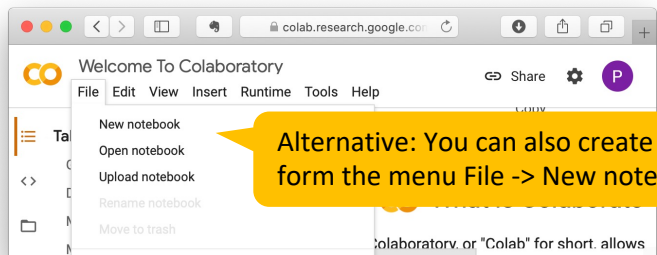
# Create your first Jupyter notebook on Google Colab



**Step 1:** Go to [colab.google.com](https://colab.google.com) and sign in with a google account.



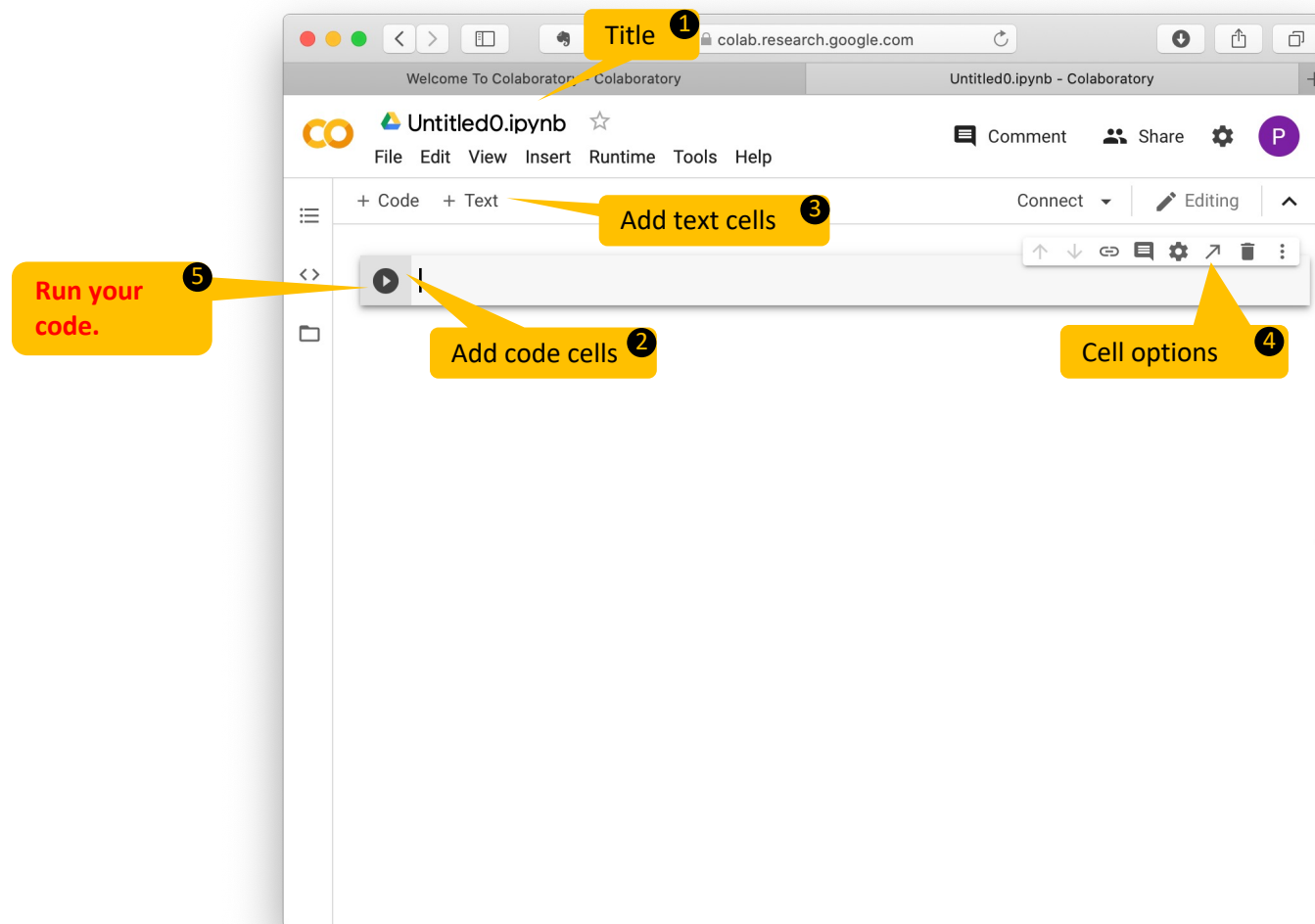
**Step 2:** Click on "NEW NOTEBOOK"



**Alternative:** You can also create new form the menu File -> New notebook



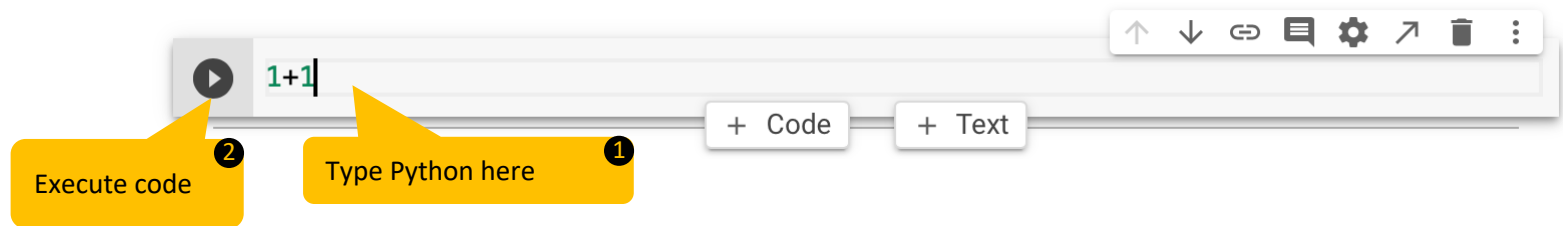
# Main components of a Google Colab notebook



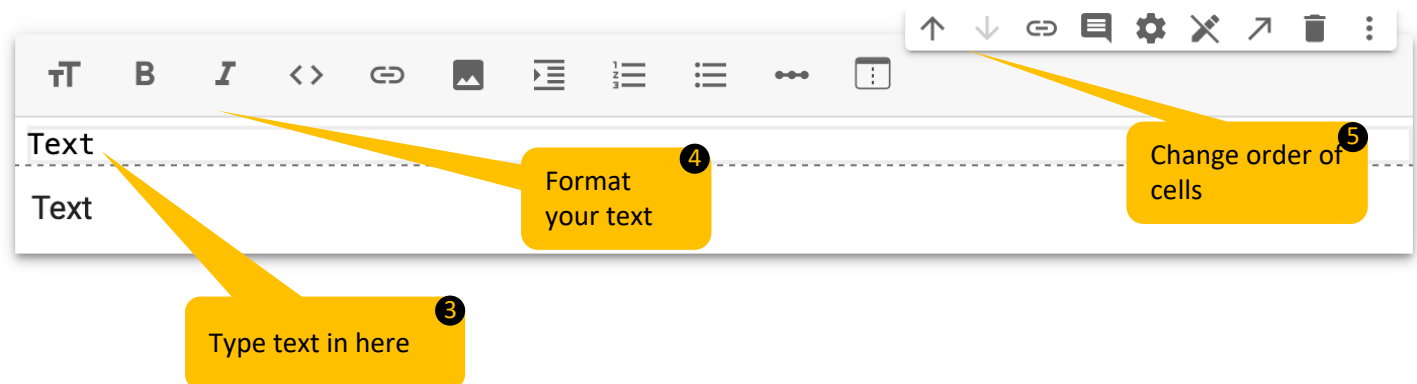
# Elements of a Jupyter Notebook (which you can e.g. use through the service “Google Colab”)

Jupyter Notebooks have two major types of "cells" for content:

- **Code Cell**



- **Markdown Cells**



**Understanding and using code notebooks**