



Load External Data into Google Colab

- [1. Uploading file through Files explorer](#)
- [2. Uploading file using Colab `files` module](#)
- [3. Reading file from Github](#)
- [4. Cloning a Github repository](#)
- [5. Downloading files from the web using Linux `wget` command](#)

[Rename file](#)

[Save file to a specific location](#)

[Invalid HTTPS SSL certificate](#)

[Multiple files at once](#)

- [6. Accessing Google Drive by mounting it locally](#)

- [7. Loading Kaggle datasets](#)

[Conclusion](#)

1. Uploading file through Files explorer

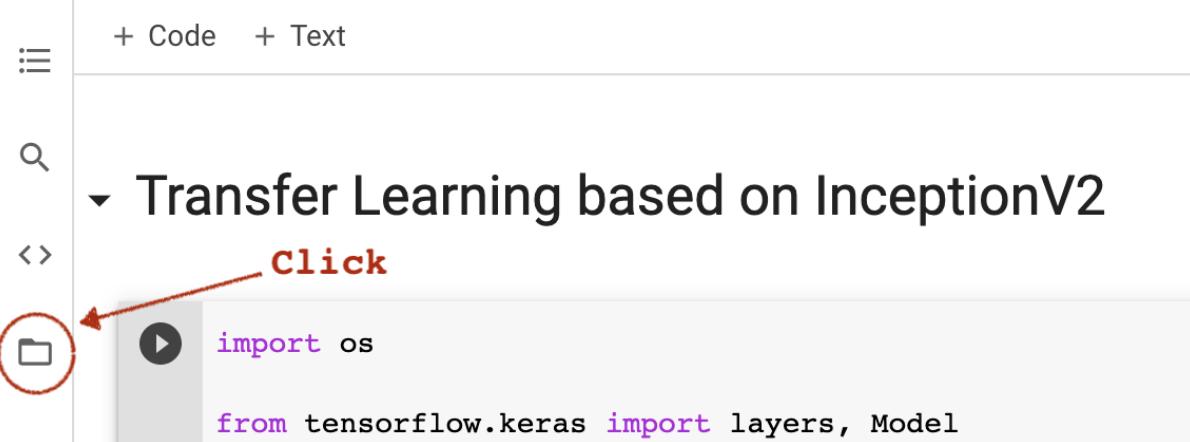
You can use the upload option at the top of the Files explorer to upload any file(s) from your local machine to Google Colab.

Here is what you need to do:

Step 1: Click the **Files** icon to open the “Files explorer” pane

CO Course 2 - Week 3 (Transfer Learning).ipynb ☆

File Edit View Insert Runtime Tools Help Last edited on April 18



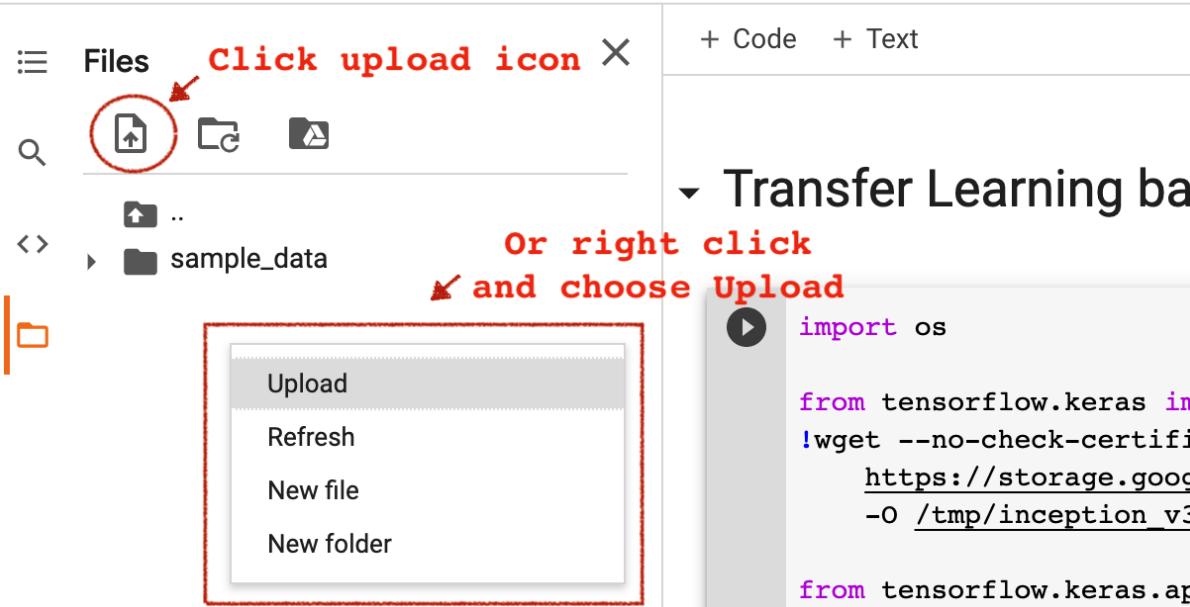
```
+ Code + Text  
Q  
<>  
▼ Transfer Learning based on InceptionV2  
Click  
import os  
from tensorflow.keras import layers, Model
```

Click Files icon (Image by author)

Step 2: Click the **upload icon** and select the file(s) you wish to upload from the “File Upload” dialog window.

CO Course 2 - Week 3 (Transfer Learning).ipynb ☆

File Edit View Insert Runtime Tools Help Last edited on April 18



Files Click upload icon X

sample_data

Or right click ↵ and choose Upload

```
+ Code + Text  
Q  
<>  
▼ Transfer Learning ba  
Import os  
from tensorflow.keras im  
!wget --no-check-certifi  
https://storage.goog  
-O /tmp/inception_v3  
from tensorflow.keras.ap
```

(Image by author)

Step 3: Once the upload is complete, you can read the file as you would normally. For instance, `pd.read_csv('Salary_Data.csv')`

The screenshot shows the Google Colab interface. On the left, there's a sidebar titled "Files" with a search bar and icons for upload, folder, and refresh. Below it, a tree view shows a folder "sample_data" containing a file "Salary_Data.csv", which is highlighted with a red border. On the right, there are two tabs: "+ Code" and "+ Text". Under the "+ Code" tab, a code cell contains the following Python code:

```
[2] import pandas as pd  
df = pd.read_csv('Salary_Data.csv')  
df.head()
```

Below the code cell, the output shows the first five rows of a DataFrame:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

(Image by author)

2. Uploading file using Colab `files` module

Instead of clicking the GUI, you can also use Python code to upload files. You can import `files` module from `google.colab`. Then call `upload()` to launch a “File Upload” dialog and select the file(s) you wish to upload.

```
from google.colab import files  
uploaded = files.upload()
```

Choose files

No file chosen

Cancel upload

File Upload dialog

Once the upload is complete, your file(s) should appear in “Files explorer” and you can read the file as you would normally.

The screenshot shows the Google Colab interface. On the left, there's a sidebar titled "Files" with a search bar and icons for upload, download, and refresh. Below it is a tree view of files: "sample_data" contains "Pokemon.csv" and "Salary_Data.csv". The main area has tabs for "+ Code" and "+ Text". A code cell at [4] shows the command to upload files from Google Drive:

```
[4] from google.colab import files
uploaded = files.upload()
```

A message box indicates the upload of two files: "Pokemon.csv" and "Salary_Data.csv". Below this, another code cell at [5] imports pandas and reads the "Salary_Data.csv" file:

```
[5] import pandas as pd
df = pd.read_csv('Salary_Data.csv')
df.head()
```

The output shows the first five rows of the DataFrame:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

(Image by author)

3. Reading file from Github

One of the easiest ways to read data is through Github. Click on the dataset in the Github repository, then click the “Raw” button.

The screenshot shows a raw data link from a Github repository. At the top, it says "Executable File | 892 lines (892 sloc) | 59.8 KB". Below is a search bar and a "Raw" button highlighted with an orange box. The table below shows the first few rows of the Titanic dataset:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
1	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25	
2	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599	71.2833	C85

(Image by author)

Copy the raw data link and pass it to the function that can take a URL. For instance, pass a raw CSV URL to Pandas `read_csv()`:

```
import pandas as pd
df = pd.read_csv('https://raw.githubusercontent.com/BindiChen/machine-learning/master/data-analysis/001-pandad-pipe-function/data/train.csv')
```

4. Cloning a Github repository

You can also clone a Github repository into your Colab environment in the same way as you would in your local machine, using `git clone`.

```
!git clone https://github.com/BindiChen/machine-learning.git
```

Once the repository is cloned, you should be able to see its contents in “Files explorer” and you can simply read the file as you would normally.

The screenshot shows the Google Colab interface. On the left, the 'Files' sidebar displays a directory structure for a notebook named 'Untitled0.ipynb'. The 'machine-learning' folder contains several subfolders and files, including 'data-analysis', 'tensorflow2', '001-googles-7-steps-of-ma...', '002-3-ways-to-build-machin...', '003-model-regularization', '004-batch-norm', '005-early-stopping', '006-learning-rate-schedul...', '007-keras-callback', 'model_checkpoints', 'keras-callbacks.ipynb', 'results.csv', '008-keras-custom-callback', '010-popular-activation-fun...', '011-relu', 'web-scraping', and 'README.md'. A green box highlights the 'sample_data' file. On the right, a code cell shows the command `!git clone https://github.com/BindiChen/machine-learning.git`. Below it, the output of the command is displayed, showing the cloning process and the resulting directory structure. Another code cell below shows the command `import pandas as pd` followed by `df = pd.read_csv('machine-learning/tensorflow2/007-keras-callback/results.csv')` and `df.head()`. The resulting DataFrame is shown as a table:

	epoch	accuracy	loss	val_accuracy	val_loss
0	0	0.540375	1.632193	0.6285	1.190760
1	1	0.688250	0.999455	0.6965	0.903381
2	2	0.728125	0.818999	0.7280	0.791272
3	3	0.755125	0.733289	0.7595	0.725264
4	4	0.772500	0.678768	0.7810	0.681306

git clone and read the file in Colab (Image by author)

5. Downloading files from the web using Linux `wget` command

Since Google Colab lets you do everything which you can in a locally hosted Jupyter Notebook, you can also use Linux shell command like `ls`, `dir`, `pwd`, `cd` etc using `!`.

Among those available Linux commands, the `wget` allows you to download files using **HTTP**, **HTTPS**, and **FTP** protocols.

In its simplest form, when used without any option, `wget` will download the resource specified in the URL to the current directory, for instance:

A screenshot of a Google Colab notebook titled "Untitled0.ipynb". The left sidebar shows a "Files" section with a "sample_data" folder containing a "cats_and_dogs_filtered.zip" file. The main code editor cell contains the command `!wget https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip`. A green arrow points from the "sample_data" folder in the sidebar to the URL in the code cell. The output of the command is displayed below the code cell, showing the progress of the download:

```
[1] !wget https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip
--2021-05-15 13:47:10-- https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip
Resolving storage.googleapis.com (storage.googleapis.com)... 142.250.97.128, 64.233.170.128, 108.177.11.128, ...
Connecting to storage.googleapis.com (storage.googleapis.com)|142.250.97.128|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 68606236 (65M) [application/zip]
Saving to: 'cats_and_dogs_filtered.zip'

cats_and_dogs_filte 100%[=====] 65.43M 114MB/s in 0.6s
2021-05-15 13:47:10 (114 MB/s) - 'cats_and_dogs_filtered.zip' saved [68606236/68606236]
```

wget in Colab (Image by author)

Rename file

Sometimes, you may want to save the downloaded file under a different name. To do that, simply pass the `-O` option followed by the new name:

```
!wget https://example.com/cats_and_dogs_filtered.zip \
-O new_cats_and_dogs_filtered.zip
```

Save file to a specific location

By default, `wget` will save files in the current working directory. To save the file to a specific location, use the `-P` option:

```
!wget https://example.com/cats_and_dogs_filtered.zip \
-P /tmp/
```

Invalid HTTPS SSL certificate

If you want to download a file over HTTPS from a host that has an invalid SSL certificate, you can pass the `--no-check-certificate` option:

```
!wget https://example.com/cats_and_dogs_filtered.zip \
--no-check-certificate
```

Multiple files at once

If you want to download multiple files at once, use the `-i` option followed by the path to a file containing a list of the URLs to be downloaded. Each URL needs to be on a separate line.

```
!wget -i dataset-urls.txt
```

The following is an example shows **dataset-urls.txt**:

```
http://example-1.com/dataset.zip  
https://example-2.com/train.csv  
http://example-3.com/test.csv
```

6. Accessing Google Drive by mounting it locally

You can use the `drive` module from `google.colab` to mount your Google Drive to Colab.

```
from google.colab import drive  
drive.mount('/content/drive')
```

Executing the above statement, you will be provided an authentication link and a text box to enter your authorization code.

A screenshot of a Jupyter Notebook cell. The code cell contains the following Python code:

```
from google.colab import drive  
drive.mount('/content/drive')
```

Below the code cell, there is a message: "Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk". Below that, there is a text input field with the placeholder "Enter your authorization code:".

Click the authentication link and follow the steps to generate your authorization code. Copy the code displayed and paste it into the text box as shown above. Once it is mounted, you should get a message like:

```
Mounted at /content/drive
```

After that, you should be able to explore the contents via “Files explorer” and read the data as you would normally.

The screenshot shows the Google Colab interface. On the left, there's a sidebar titled "Files" showing a file tree. A green box highlights the "MyDrive" folder, which contains "Colab Notebooks", "Daughter App", "Other", "Easter Trip.gmap", "How to get started with Dr...", "PWA workshop.gdoc", "Salary_Data.csv" (which has an orange arrow pointing to it), and "To learn and add.gsheet". Below this is a "sample_data" folder. On the right, a code cell at index [4] contains the following Python code:

```
[4] import pandas as pd
df = pd.read_csv('drive/MyDrive/Salary_Data.csv')
df.head()
```

The output of this code is a table:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

Finally, to unmount your Google Drive:

```
drive.flush_and_unmount()
```

7. Loading Kaggle datasets

It is possible to download any dataset seamlessly from Kaggle into your Google Colab. Here is what you need to do:

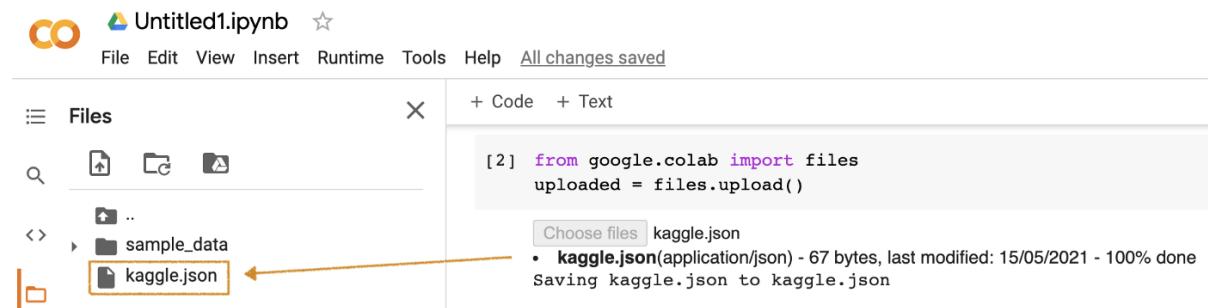
Step 1: Download your Kaggle API Token: Go to Account and scroll down to the **API** section.

The screenshot shows the Kaggle API section. At the top, there's a navigation bar with "Home", "Competitions", "Datasets", "Code", "Discussion", "Followers", "Notifications", "Account" (which is highlighted with an orange box), and "Edit Public Profile". Below this is a section titled "API". It contains the following text: "Using Kaggle's beta API, you can interact with Competitions and Datasets to download data, make submissions, and more via the command line. [Read the docs](#)". There are two buttons at the bottom: "Create New API Token" (which is highlighted with an orange box) and "Expire API Token".

Generate Kaggle API token (Image by author)

By clicking “Create New API Token”, a **kaggle.json** file will be generated and downloaded to your local machine.

Step 2: Upload **kaggle.json** to your Colab project: for instance, you can import `files` module from `google.colab`, and call `upload()` to launch a File Upload dialog and select the kaggle.json from your local machine.



Upload kaggle.json (Image by author)

Step 3: Update `KAGGLE_CONFIG_DIR` path to the current working directory. You can run `!pwd` to get the current working directory and assign the value to `os.environ['KAGGLE_CONFIG_DIR']`:

```
[3] !pwd
```

```
/content
```

```
[4] import os  
os.environ['KAGGLE_CONFIG_DIR'] = "/content"
```

Configure KAGGLE_CONFIG_DIR (Image by author)

Step 4: Finally, you should be able to run the following Kaggle API to download datasets:

```
!kaggle competitions download -c titanic!kaggle datasets download -d alexanderbader/forbes-billionaires-2021-30
```

The screenshot shows a Google Colab notebook titled "Untitled1.ipynb". The left sidebar displays a file tree with "sample_data", "gender_submission.csv", and "kaggle.json". The main area contains the following code:

```
[2] from google.colab import files
uploaded = files.upload()

[3] !pwd
/content

[4] import os
os.environ['KAGGLE_CONFIG_DIR'] = "/content"

[ ] !kaggle competitions download -c titanic
```

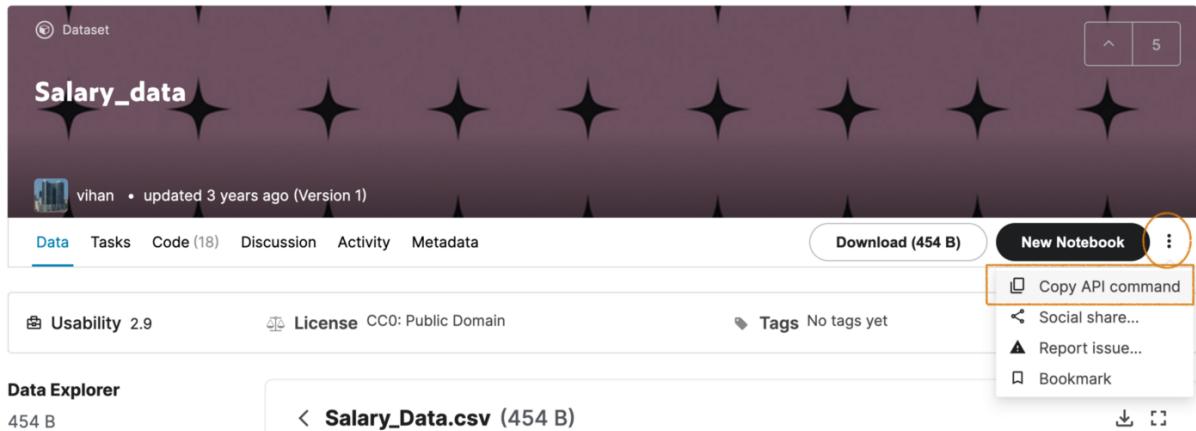
Download Kaggle Dataset (Image by author)

Note for the competition dataset, the Kaggle API should be available under the Data tab

The screenshot shows the Kaggle competition page for "Titanic - Machine Learning from Disaster". The top navigation bar includes "Overview", "Data" (which is highlighted with an orange border), "Code", "Discussion", "Leaderboard", "Rules", and "Team". Below the navigation, there's a "My Submissions" button and a "Submit Predictions" button. The main content area has a heading "Data Description" and a section titled "Overview". It states: "The data has been split into two groups:" followed by a bulleted list: "• training set (train.csv)" and "• test set (test.csv)". A note below says: "The training set should be used to build your machine learning models. For the training set, we provide the outcome (also known as the "ground truth") for each passenger. Your model will be based on "features" like passengers' gender and class. You can also use [feature engineering](#) to create new features." At the bottom, there's a command line input field containing the command: "kaggle competitions download -c titanic".

Retrieve Kaggle API from competition dataset (Image by author)

For the general dataset, the Kaggle API can be accessed as follows:



Retrieve Kaggle API from a general dataset (Image by author)

Conclusion

Google Colab is a great tool for individuals who want to take advantage of the capabilities of high-end computing resources (like GPUs, TPUs) without being restricted by their price.

In this article, we have gone through most of the ways you can improve your Google Colab experience by loading external data into Google Colab. I hope this article will help you to save time in learning Colab and Data Analysis.

Thanks for reading. Stay tuned if you are interested in the practical aspect of machine learning.