**School of Chemical and Material Engineering (SCME)**

**CS-117 Application of ICT**

**Class: Chemical & Material Engineering**

**CLO-4 Analyze and evaluate different software applications.**

**Lab 13a: Data loading in Google Colab**

**Course Instructor’s: Dr Sara Shakeel Qureshi**

**Lab Engineer: Farrukh Zain ul Abideen**

**Lab 13a: Data loading in Google Colab**

**Introduction**

The purpose of this lab is to revisit the basic concepts of python and how to use google colab for data loading.

**Objectives**

Objective of this lab is to revisit the basic concepts of python and how to use google colab for data loading.

**Tools/Software Requirement**

Google colab, jupyetr notebook

**Description**

**Colab** (short for **Colaboratory**) is a free platform from Google that allows users to code in Python. Colab is essentially the Google version of a Jupyter Notebook. Some of the advantages of Colab over Jupyter include zero configuration, free access to GPUs & CPUs, and seamless sharing of code.

More and more people are using Colab to take the advantage of the high-end computing resources without being restricted by their price. Loading data is the first step in any data science project. Often, loading data into Colab require some extra setups or coding. In this lab, we’ll learn the 7 common ways to load external data into Google Colab, i.e.,

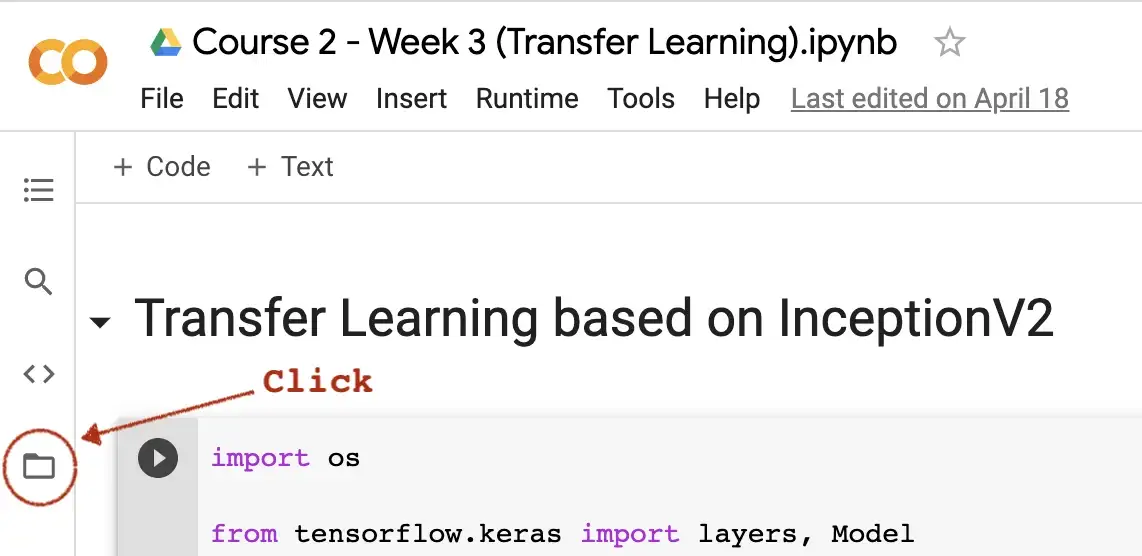
1. Uploading file through Files explorer
2. Uploading file using files module
3. Reading a file from Github
4. Cloning a Github Repository
5. Downloading files using Linux wget command
6. Accessing Google Drive by mounting it locally
7. Loading Kaggle Datasets

**1. Uploading file through Files explorer**

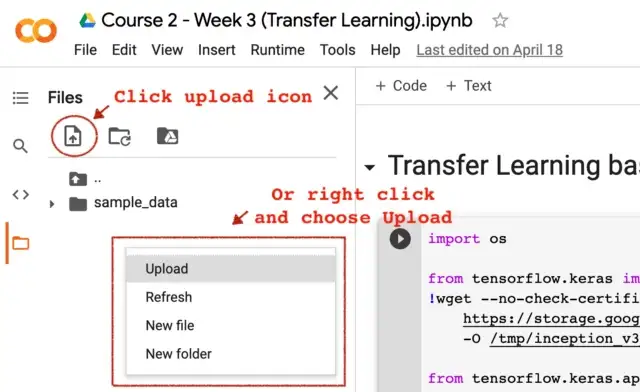
You can use the upload option at the top of the Files explorer to upload any file(s) from your local machine to Google Colab.

Here is what you need to do:

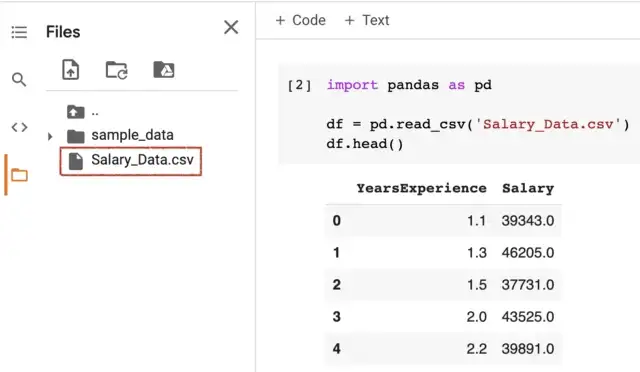
**Step 1:** Click the **Files** icon to open the “Files explorer” pane Click Files icon



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



**Step 3:** Once the upload is complete, you can read the file as you would normally. For instance, pd.read\_csv('Salary\_Data.csv')



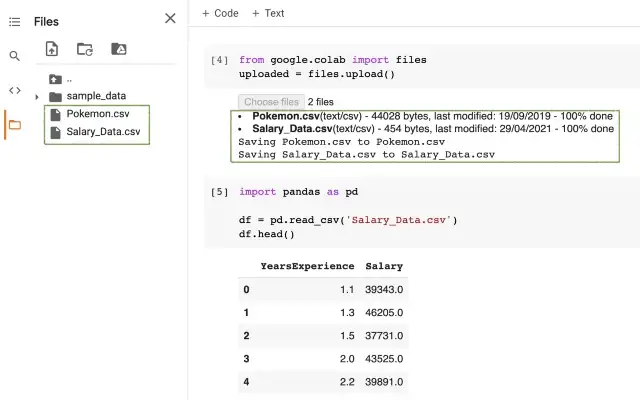
**2. Uploading file using Colab**files**module**

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()



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



**3. Reading file from Github**

One of the easiest ways to read data is through Github. Click on the [dataset](https://github.com/BindiChen/machine-learning/blob/master/data-analysis/001-pandad-pipe-function/data/train.csv) in the Github repository, then click the “**Raw**” button.



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

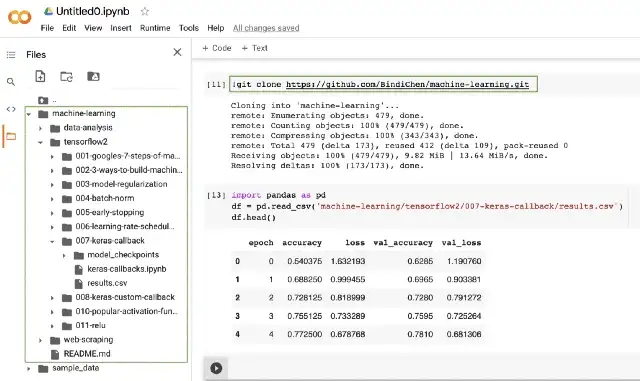
df = pd.read\_csv('<https://raw.githubusercontent.com/BindiChen/machine-learning/master/data-analysis/001-pandad-pipe-function/data/train.csv>')

**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.



**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:



**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

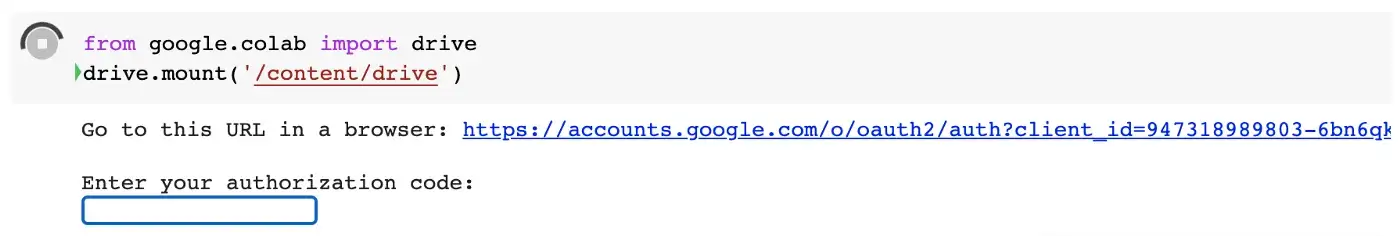
**6. Accessing Google Drive by mounting it locally**

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')**

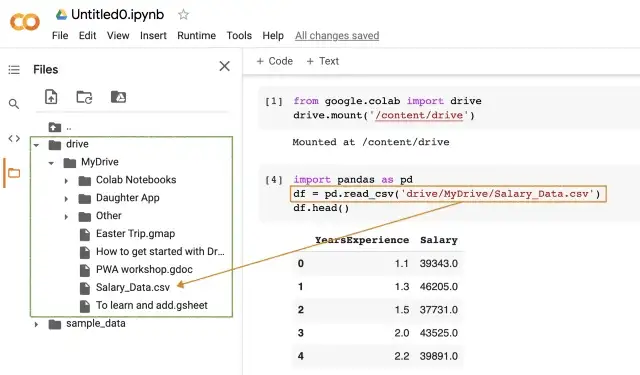
Executing the above statement, you will be provided an authentication link and a text box to enter your authorization code.



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.



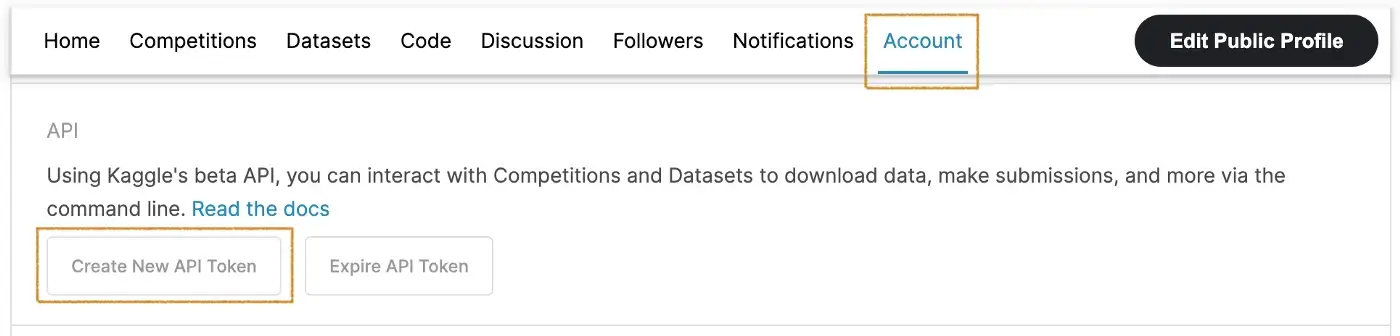
Finally, to unmount your Google Drive:

drive.flush\_and\_unmount()

**7. Loading Kaggle datasets**

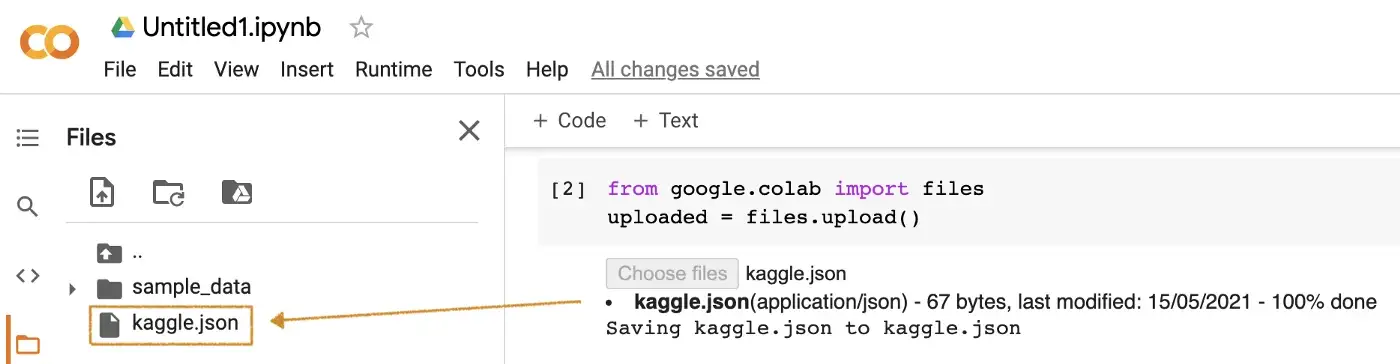
It is possible to download any dataset seamlessly from Kaggle into your Google Colab. Here is what you need to do:

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



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.



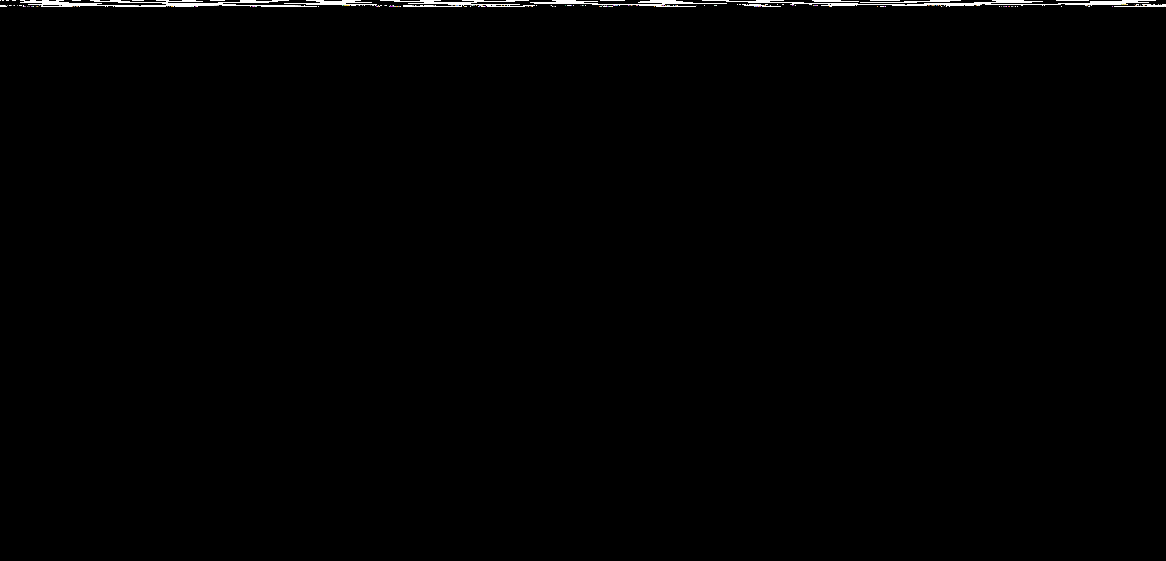
**Step 3:** Update KAGGLE\_CONFIG\_DIR path to the current working directory. You can run !pwd to get the current working directory and assign the value to os.environ['KAGGLE\_CONFIG\_DIR'] :



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

!kaggle competitions download -c titanic

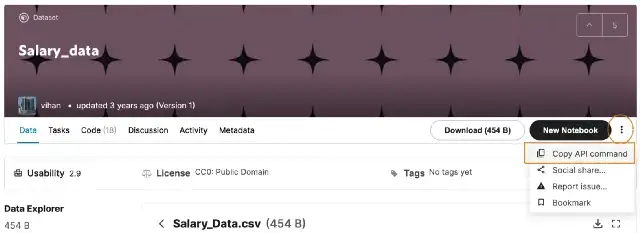
!kaggle datasets download -d alexanderbader/forbes-billionaires-2021-30



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



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



**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 lab, we have gone through most of the ways you can improve your Google Colab experience by loading external data into Google Colab.

Deliverable:

Submit your jupyter file showing all the methods to import data in colab before deadline.