Big Data Analytics

Isara Anantavrasilp

Lecture 13: Data Visualization with Python

Google Colab

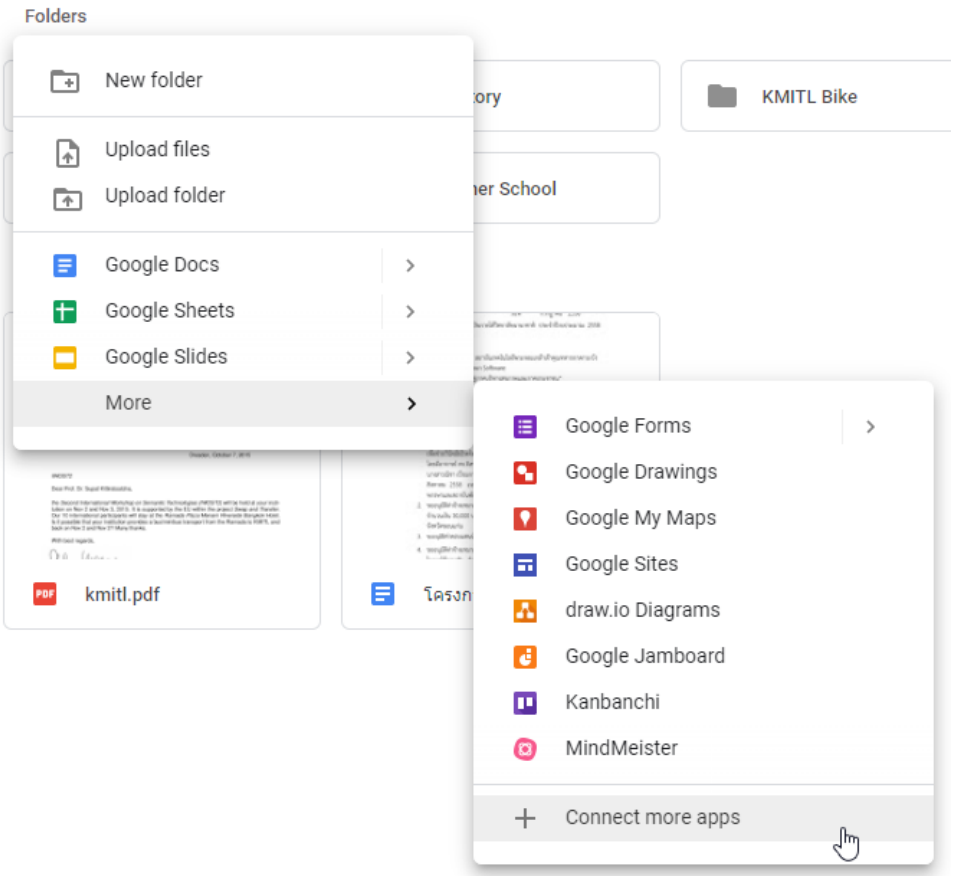
- Full name: **Google Colaboratory** (with 1 l)
- Google Colab is a platform to write/share/run codes on Google Drive
  - It is a Google Drive plugin
  - Codes are saved in Drive
  - Files in Drive can also be accessed.
- It based on Jupyter Notebook
- Supported languages
  - Python 2.7
  - Python 3.6
  - R and Scala are not supported yet

Google Colab vs Jupyter Notebook

- Colab is based on Jupyter Notebook
  - Essentially, it is a shared Notebook on Google Drive
  - Created in collaboration with Jupyter developers
  - Run and display results like Notebook
- **Supports GPU computations**
- You do not need to install Jupyter and Python on your machine, simply install Colab plugin in Google Drive

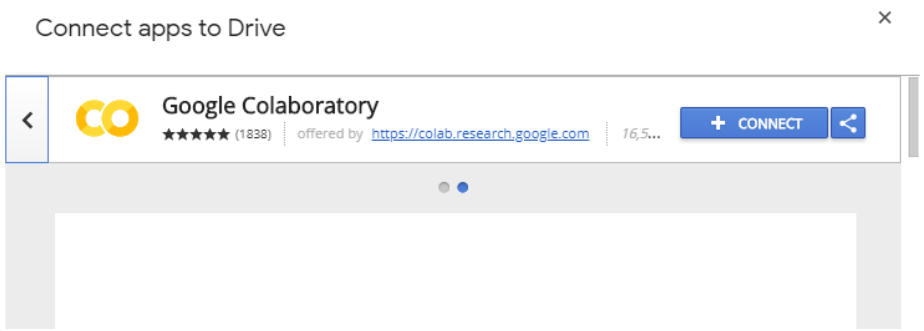
Installing Colab

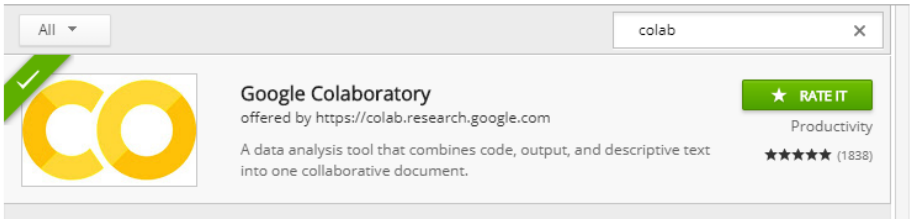
- Open your Google Drive
- Create a folder to store your notebook
- Then right click anywhere
- Then select More > Connect more apps..

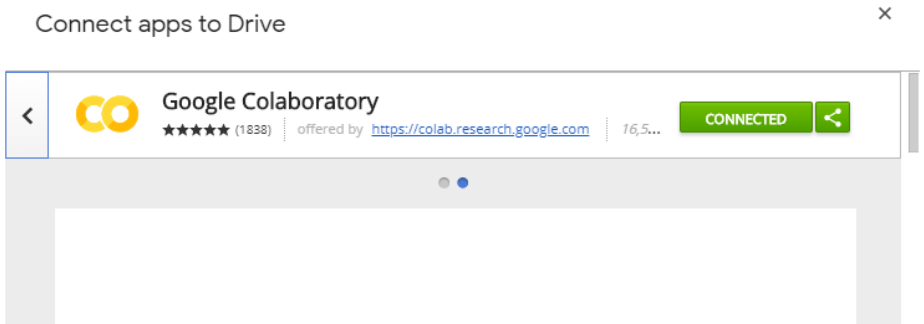


Installing Colab

- Search “Colab”
- Then click “CONNECT”
- If you have already connected with Colab, it will show “RATE IT” instead
- If you click further, you will see “CONNECTED” button

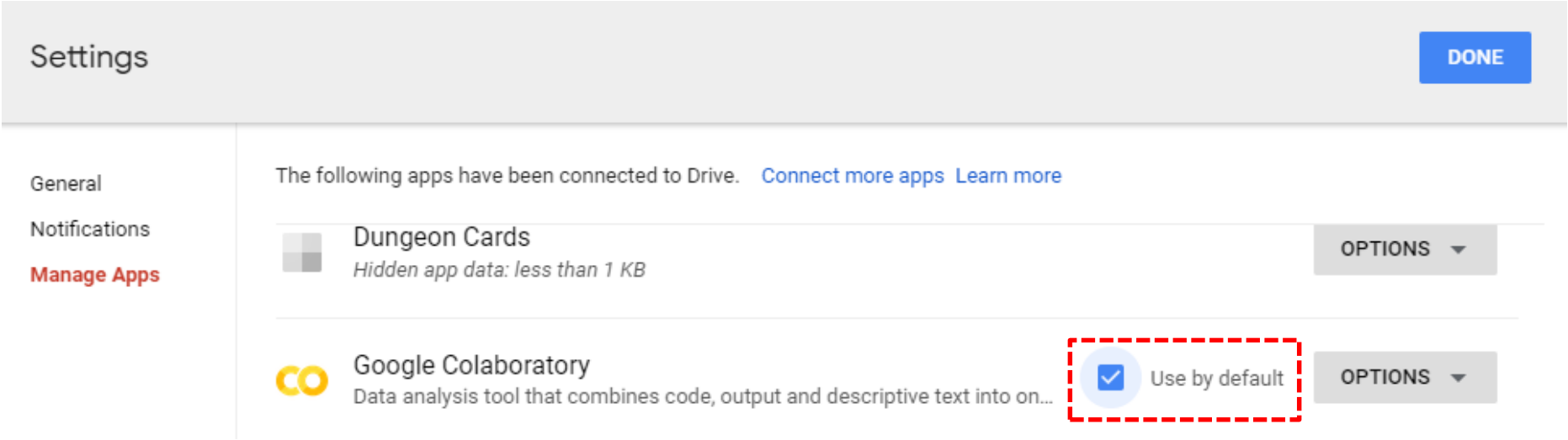






Installing Colab

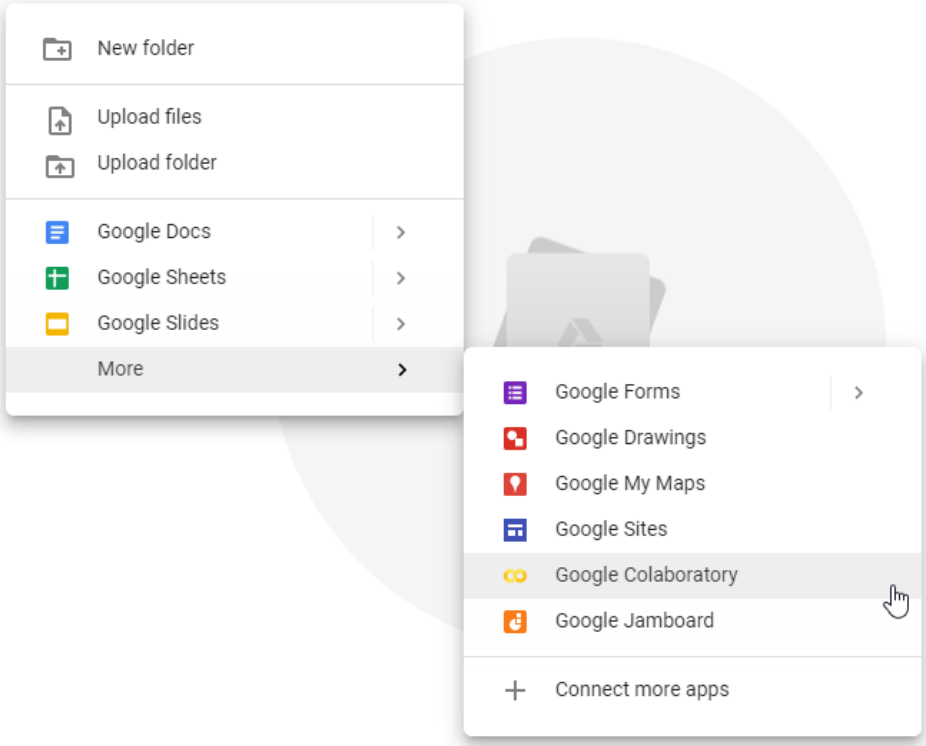
- By default Colab will be shown as an option for you to create new file (unlike Google Docs or Sheets)
- Go to Setting by clicking ⚙ า
- Go to Manage Apps
- Scroll down to Google Colaboratory
- Then select “Use by default”



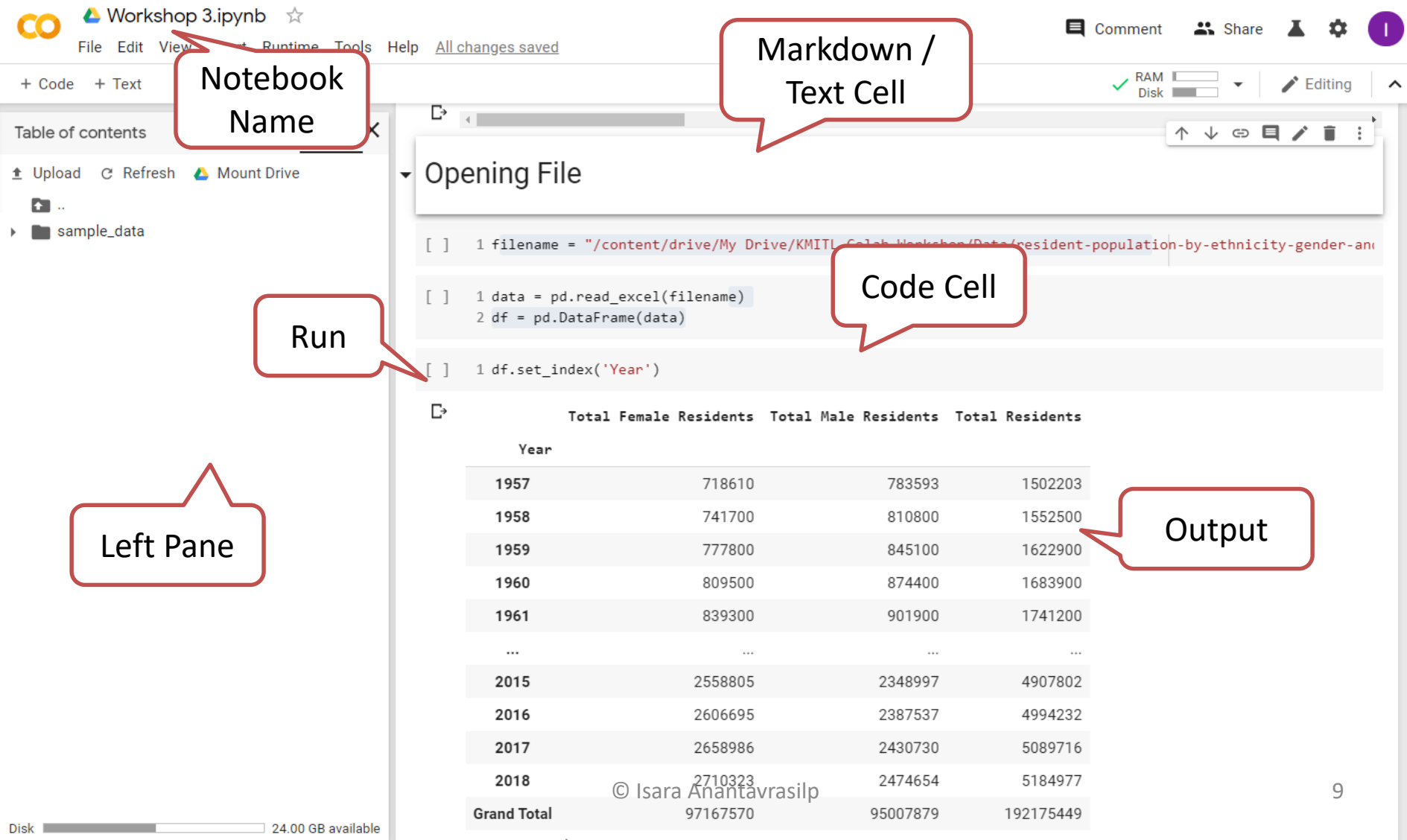
Starting Colab

- Go to your project folder
  - Create one if you don't have one yet
- Right-click anywhere
- Select

  More > Google Colab
- Drive will start new Colab window



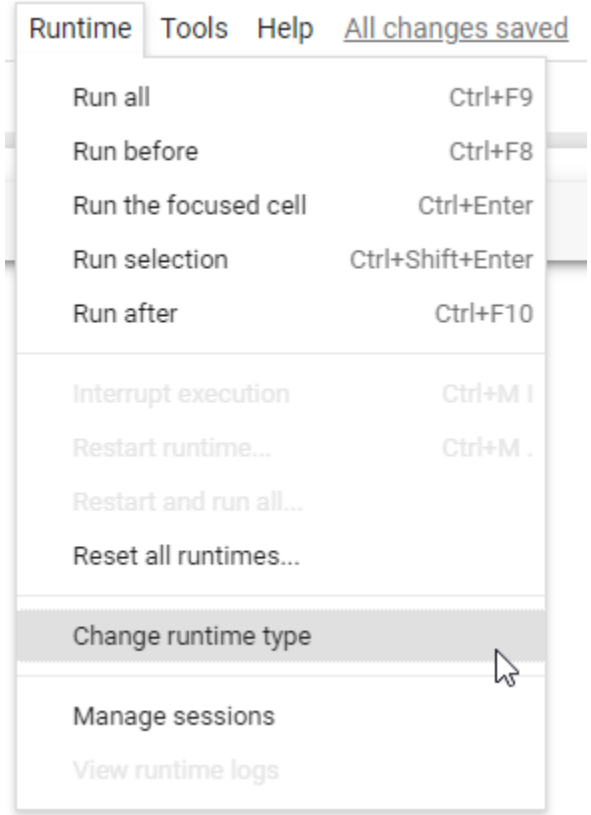
Colab UI

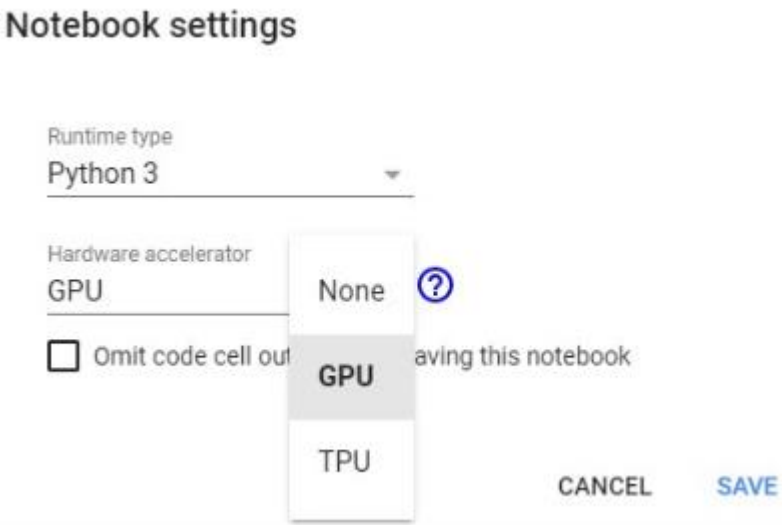


	Total Female Residents	Total Male Residents	Total Residents
Year			
1957	718610	783593	1502203
1958	741700	810800	1552500
1959	777800	845100	1622900
1960	809500	874400	1683900
1961	839300	901900	1741200
...	...	...	...
2015	2558805	2348997	4907802
2016	2606695	2387537	4994232
2017	2658986	2430730	5089716
2018	2710323	2474654	5184977
Grand Total	97167570	95007879	192175449

Runtime Type

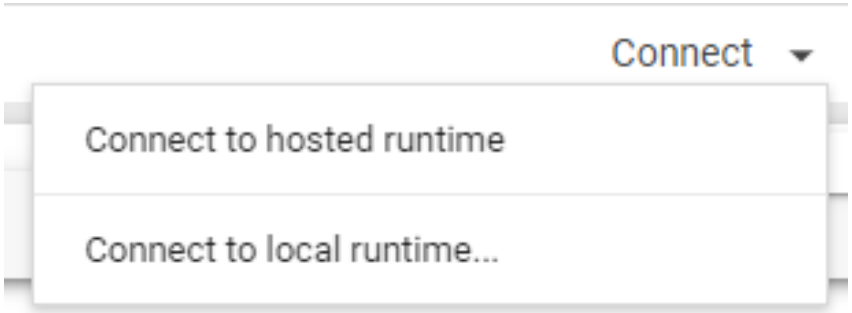
- A bit of configuration: You should select how your project is compiled and run
- Open menu Runtime
- > Change runtime type
- Select Runtime type: Python 3
- Hardware accelerator: GPU / TPU





Server and Local Runtime

- Google Colab runs mainly on server side (i.e. hosted by Google)
- You can also connect to your local interpreter (runtime)
  - You can manage your own resources
  - Access to sensitive data
  - Use Drive and Colab only to store and write codes
- You can select runtime by clicking Connect
  - Connect to hosted runtime
  - Connect to local runtime
- We will focus only on hosted runtime



Pandas

- Pandas is an opensource data analysis library for Python
  - Built on top of **NumPy** package
  - Designed to manipulate data, e.g., loading, cleaning
- Data prepared by Pandas can be used to for further analysis
  - Stats analysis: **SciPy**
  - Plotting: **Matplotlib** or **Datashader**
  - Machine Learning: **Scikit-learn**

Series and DataFrames

- Pandas use two main components to store data
  - **Series**: Series of values of the same type
  - **DataFrame**: Multi-dimensional table created by combining multiple series. Essentially, a series is a column in a DataFrame
- DataFrame can be created many ways e.g., from a dictionary or import and convert from other sources such as a CSV file

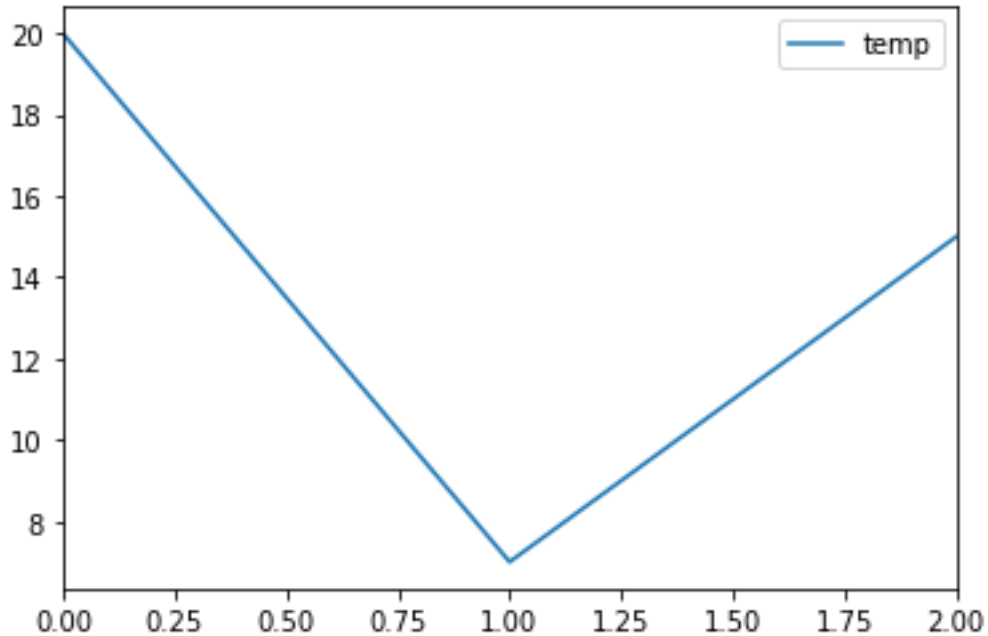
Workshop 1: DataFrame

```
import pandas as pd
data = {
    'day': ['5/7/2019', '2/7/2019', '3/7/2019'],
    'temp': [20, 7, 15],
    'event': ['cold', 'cold', 'cold']
}
df = pd.DataFrame(data)

df
print(df.shape)
print(df)
print(df['temp'].max())
```

Plot

```
df.plot()
```




```
df = df.sort_values(by="temp", ascending=True)
df.plot()
```

Workshop 1: Adding Column

- You can add column by adding a series

```
df.insert(3, "Play Tennis", ["Yes","No","Yes"])
```

- The series can be imported from external files such as CSV or Excel or it can be a result of some computation

```
df.insert(3, "Play Tennis", series)
```

Workshop 1: Removing Column

- You can drop a column by:

```
df.drop("Event", axis = 1)
```

- axis = 1 explicitly denotes that we want to drop the column, not a row.

- You can also drop multiple columns at once

```
df.drop(["Event","Play Tennis"], axis = 1)
```

Workshop 2: Accessing Files

- Colab can access data files from your Drive
- Open menu on the left
- Then select Files > Mount Drive
- Colab will add the following lines to your code

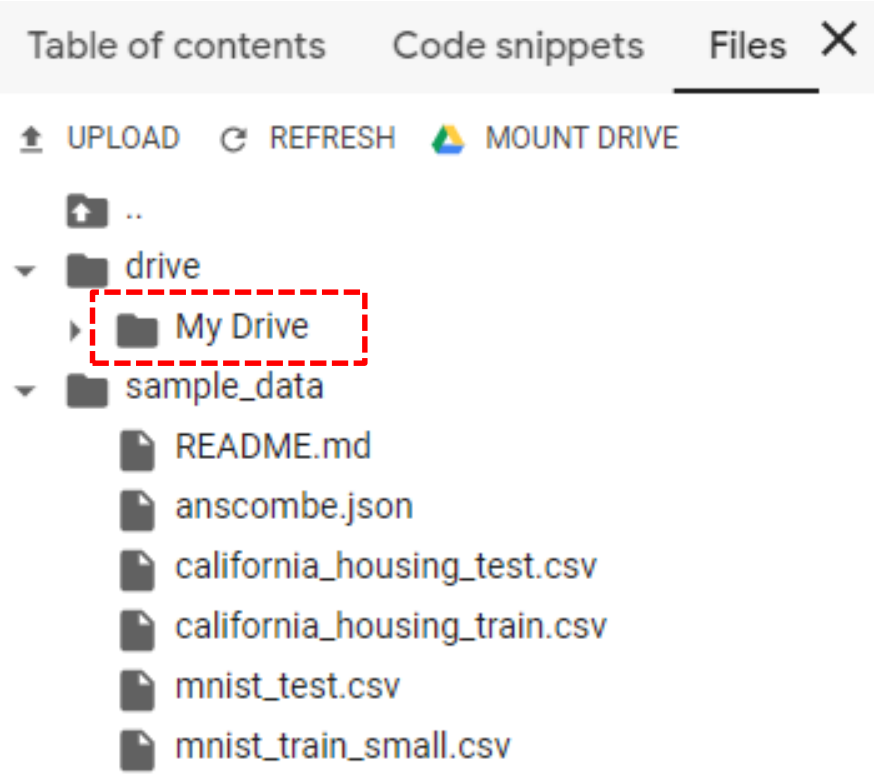
```
from google.colab import drive
drive.mount('/content/drive')
```

- Run and following instructions

Accessing files

- You will see your Drive mounted at the mount point

`/content/drive`



Working with Files

- You can work with files similar to your local workspace
- List files in folder:

```
!ls "/content/drive/My Drive/”
```

- Download remote file:

```
!wget "https://storage.data.gov.sg/resident-population-by-ethnicity-gender-and-age-group/resident-population-by-ethnicity-gender-and-age-group.zip”
```

- Unzip:

```
!unzip "resident-population-by-ethnicity-gender-and-age-group.zip”
```

Workshop 3: Loading Data

- Download the following file and put to your Drive

https://bit.ly/2QFtPrR

- Then, open another notebook
- And open the file in the notebook

`singapore-residents-total.xlsx`

Workshop 3: Loading Data

- Then, load of the data into data frame

```
import pandas as pd

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

filename = "/content/drive/My Drive/Big Data
Analytic/Example Data/singapore-residents-total.xlsx"

data = pd.read_excel(filename)
df = pd.DataFrame(data)
df
```

Workshop 3: Data Manipulation

- See the beginning of the data

```
df.head()
```

- See the end of the data

```
df.tail()
```

- Data types

```
df.types
```

- Filtering

```
mil = df["Total Female Residents"] > 1500000
df.loc[mil]
```

Workshop 3: Basic Stats

- Sum of a row

```
df["Total Female Residents"].sum()
```

- Mean of a row

```
df["Total Female Residents"].mean()
```

- Mean of all columns

```
df.mean(axis=0)
```

- Mean of all rows

```
df.mean(axis=0)
```

- Describe me!

```
df["Total Female Residents"].describe()
```

Workshop 3: First Plot

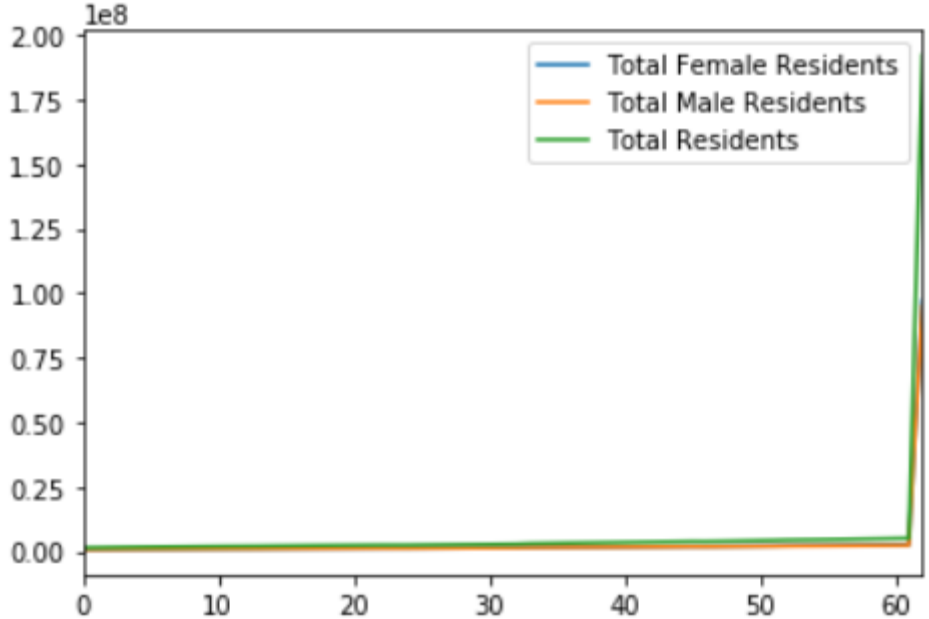
- Set index of the data to year, instead of row number

```
df.set_index('Year')
```

- Then plot the chart

```
df.plot()
```

- Why does it look like this?



Workshop 3: Removing Row

- Remove “Grand Total” row by dropping the last row

```
df.drop(df.tail(1).index,inplace=True)
```

- You can also drop the first row

```
df.drop(df.head(1).index,inplace=True)
```

- Just change 1 to any number if you want to drop more rows

inplace

- You will come across many operations in Pandas that has parameter inplace
- When inplace is set to True, it means that the change should be done in the source data frame
- If it is set to False, the changed data will have to be assign to another data frame

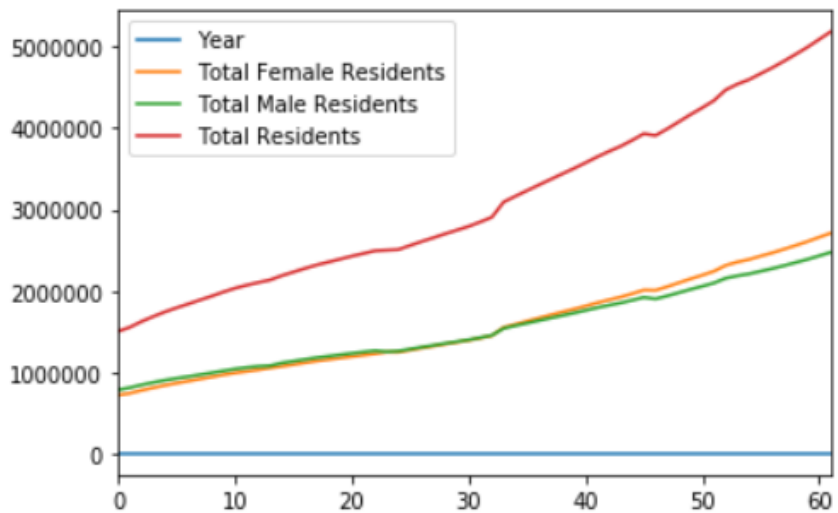
```
df2 = df.drop(df.tail(1).index,inplace=False)
```

Workshop 3: Removing Row

- Plot again:

```
df.plot()
```

- You should get

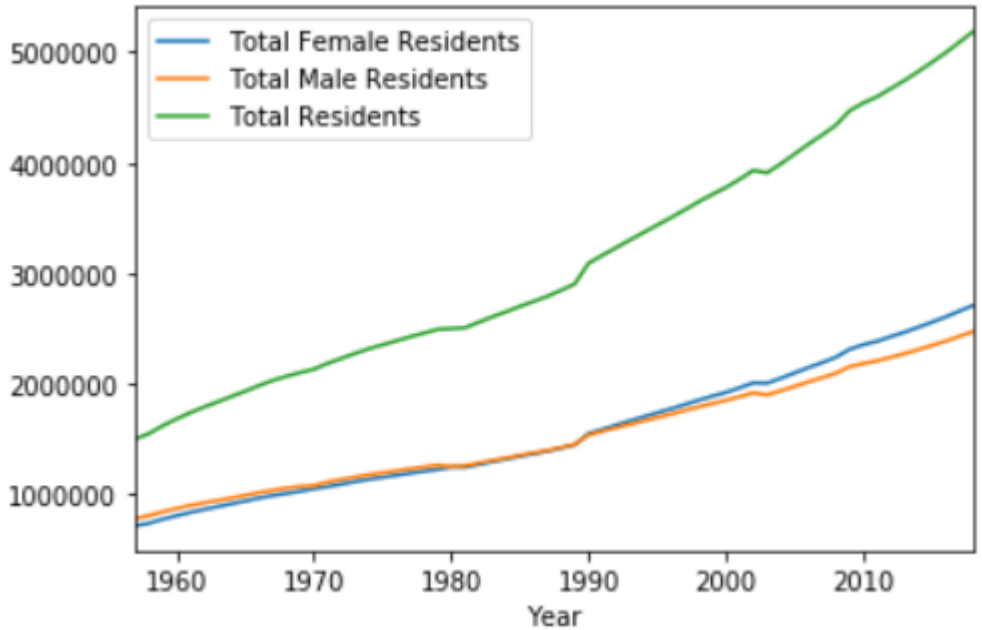


- Still not what we want. We want year in the x-axis.

Workshop 3: Plot against a column

- Specifying columns to plot:

```
df.plot(x='Year')
```

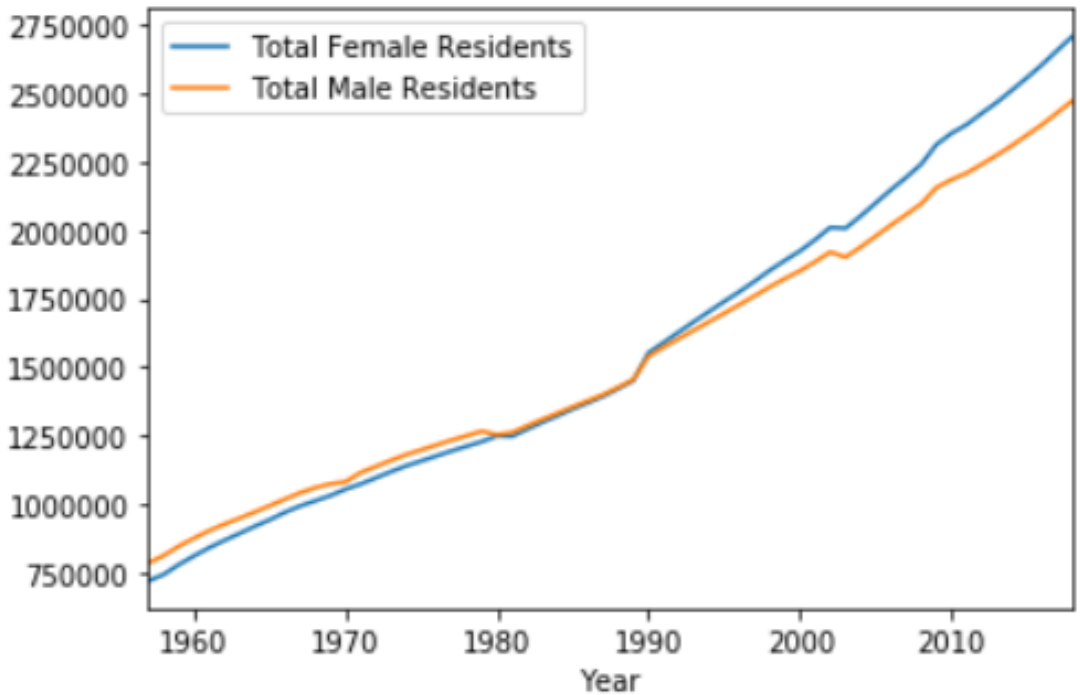


Workshop 3: Plot Selected Series

- Specifying columns to plot:

```
df.plot(x='Year')
```

- **Exercise**: Select specific series to plot



Workshop 4: KMITL Bike Data

- Download the data

  https://bit.ly/37e2pyW

- The Dataset contains locations of bike rides between April – September

- Open the following notebook

  https://bit.ly/3416dSi

More on Data Analysis

- What we have done was only an introduction
- Python provides large set of tools for data analysis
  - NumPy: ML, Scientific and numerical computation
  - Matplotlib: 2D plotting
  - Pandas: Data manipulation, basic visualization
  - Seaborn: Statistical data visualization
  - Scikit-Learn: Machine learning and data analysis
  - TensorFlow: Deep learning
  - Keras: Deep learning
- Colab and Jupyter Notebook are very good to develop, execute and share such works