HOW TO DO TRAIN TEST SPLITS

WITH PYTHON

DR. ALVIN ANG

CONTENTS

I. TRAIN TEST SPLIT THE BOSTON HOUSING DATASET.CSV

https://www.alvinang.sg/s/How_to_Do_Train_Test_Splits_with_Python_by_Dr_Alvin_Ang_v3.ipynb

0. Splitting the Boston Housing Dataset.CSV

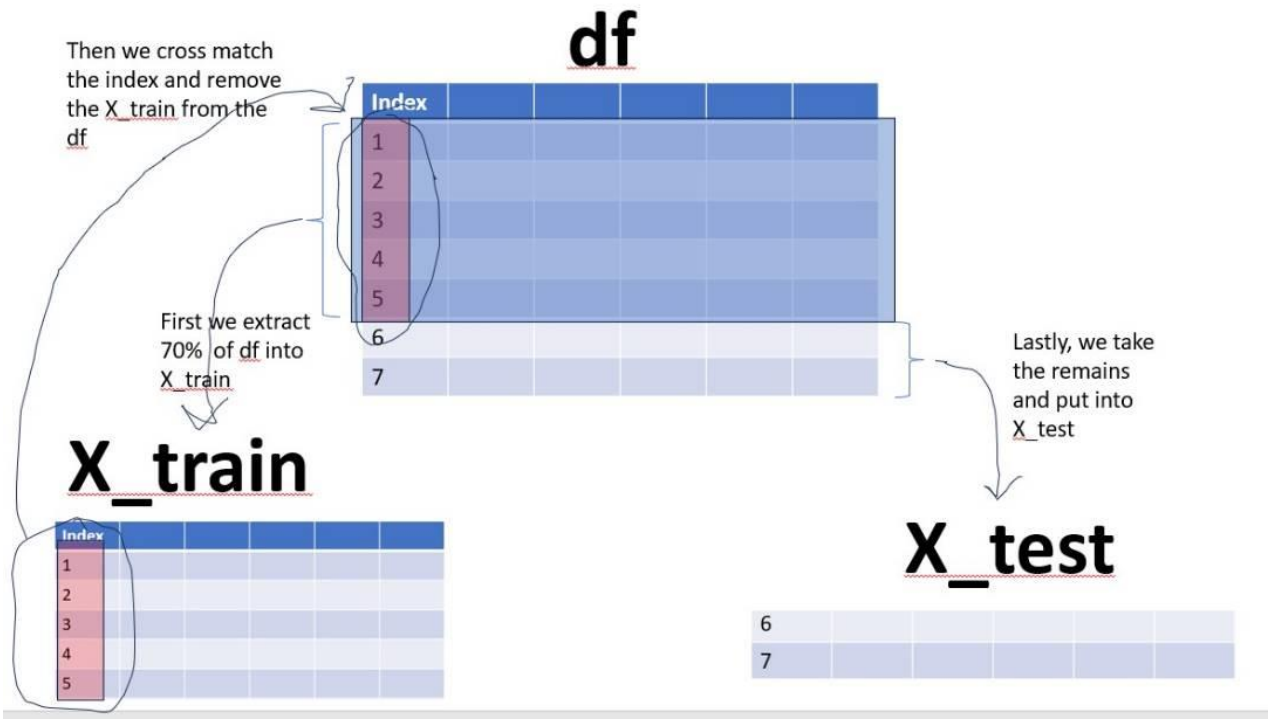
a) Loading the CSV

```
[3]  1 import pandas as pd
     2 df = pd.read_csv('https://www.alvinang.sg/s/boston_housing_data.csv')
     3 df
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	MEDV
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222.0	18.7	396.90	5.33	36.2

b) Train Test Split

```
[4]  1 #X_train will sample 70% of the df
     2 X_train = df.sample(frac=0.7)
     3
     4 #then we keep the remaining 30% and store into X_test
     5 X_test = df.drop(X_train.index)
```




```
[6]  1 y_train = X_train.pop('MEDV')
     2
     3 y_test = X_test.pop('MEDV')
```

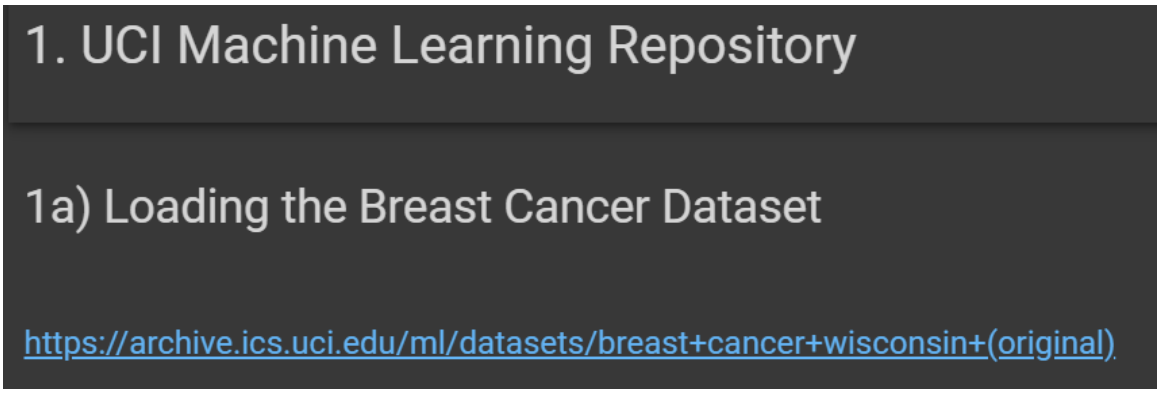
c) Normalize

```
1 mean, std = X_train.mean(), X_train.std()
2 X_train = (X_train - mean)/std
3
4 #must reuse mean and std from x_train
5 X_test = (X_test - mean)/std
```

II. UCI MACHINE LEARNING REPOSITORY

https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(original)

A. LOADING THE BREAST CANCER DATASET



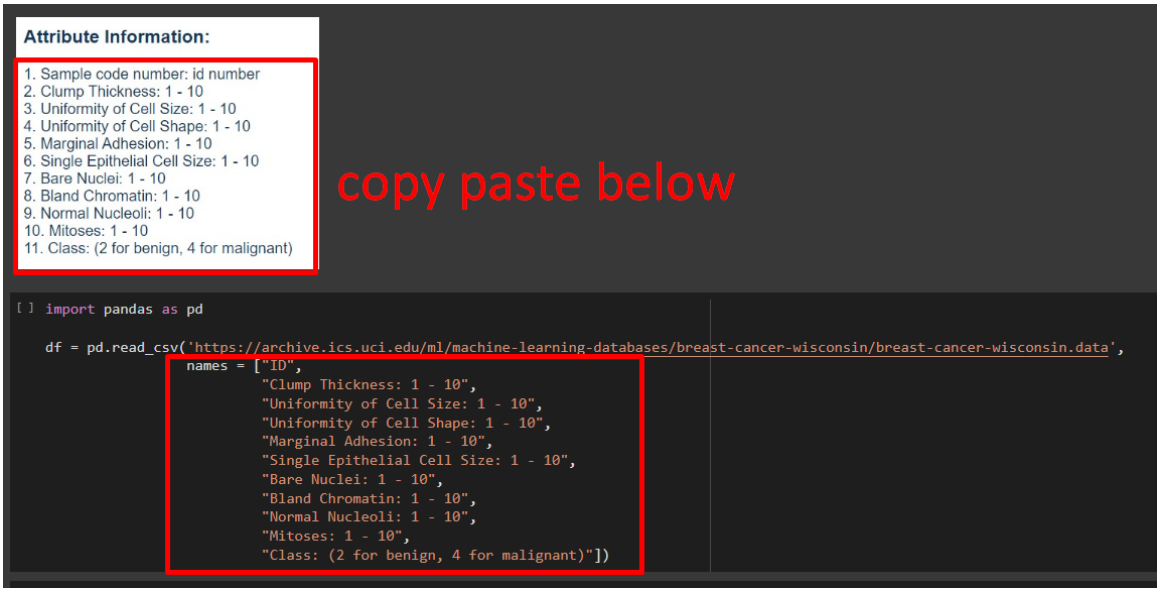
```
import pandas as pd

df = pd.read_csv 'https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data',
                 names = ["ID",
                          "Clump Thickness: 1 - 10",
                          "Uniformity of Cell Size: 1 - 10",
                          "Uniformity of Cell Shape: 1 - 10",
                          "Marginal Adhesion: 1 - 10",
                          "Single Epithelial Cell Size: 1 - 10",
                          "Bare Nuclei: 1 - 10",
                          "Bland Chromatin: 1 - 10",
                          "Normal Nucleoli: 1 - 10",
                          "Mitoses: 1 - 10",
                          "Class: (2 for benign, 4 for malignant)"])
```

copy paste the link here

Attribute Information:

1. Sample code number: id number
2. Clump Thickness: 1 - 10
3. Uniformity of Cell Size: 1 - 10
4. Uniformity of Cell Shape: 1 - 10
5. Marginal Adhesion: 1 - 10
6. Single Epithelial Cell Size: 1 - 10
7. Bare Nuclei: 1 - 10
8. Bland Chromatin: 1 - 10
9. Normal Nucleoli: 1 - 10
10. Mitoses: 1 - 10
11. Class: (2 for benign, 4 for malignant)



df

	ID	Clump Thickness: 1 - 10	Uniformity of Cell Size: 1 - 10	Uniformity of Cell Shape: 1 - 10	Marginal Adhesion: 1 - 10	Single Epithelial Cell Size: 1 - 10	Bare Nuclei: 1 - 10	Bland Chromatin: 1 - 10	Normal Nucleoli: 1 - 10	Mitoses: 1 - 10	Class: (2 for benign, 4 for malignant)
0	1000025	5	1	1	1	2	1	3	1	1	2
1	1002945	5	4	4	5	7	10	3	2	1	2
2	1015425	3	1	1	1	2	2	3	1	1	2
3	1016277	6	8	8	1	3	4	3	7	1	2
4	1017023	4	1	1	3	2	1	3	1	1	2
...	...	...	...	...	...	...	...	...	...	...	...
694	776715	3	1	1	1	3	2	1	1	1	2
695	841769	2	1	1	1	2	1	1	1	1	2
696	888820	5	10	10	3	7	3	8	10	2	4
697	897471	4	8	6	4	3	4	10	6	1	4
698	897471	4	8	8	5	4	5	10	4	1	4

699 rows × 11 columns

B. RENAMING COLUMNS

1b) Renaming Columns

```
[ ] df.rename(columns = {"Class: (2 for benign, 4 for malignant)":"Class"}, inplace = True)

    #rename the "Class" column

[ ] df["Class"] = df["Class"].map({2:0, 4:1})

    #0: benign (no cancer)
    #1: malignant (cancer)
```

df

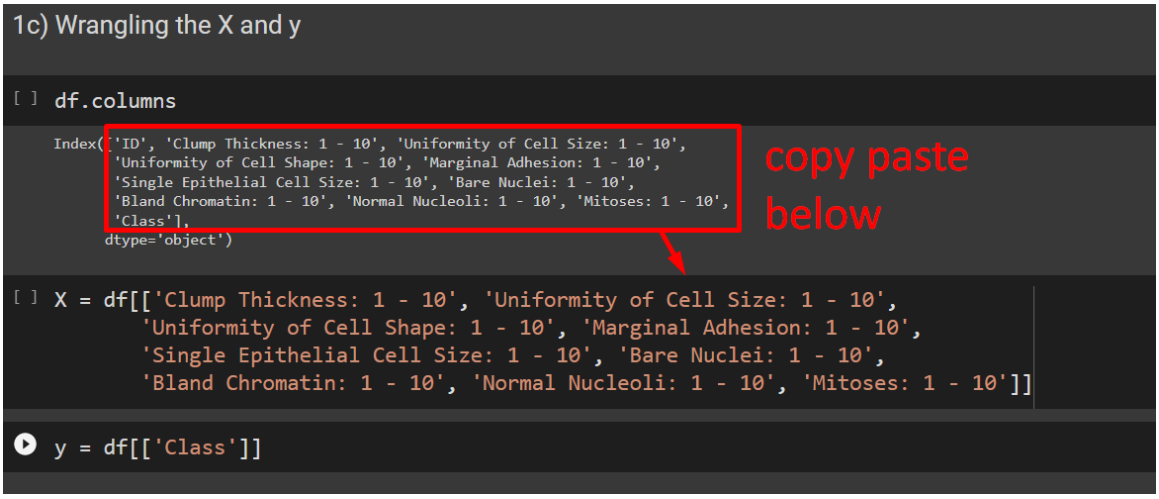
renamed and relabelled

	ID	Clump Thickness: 1 - 10	Uniformity of Cell Size: 1 - 10	Uniformity of Cell Shape: 1 - 10	Marginal Adhesion: 1 - 10	Single Epithelial Cell Size: 1 - 10	Bare Nuclei: 1 - 10	Bland Chromatin: 1 - 10	Normal Nucleoli: 1 - 10	Mitoses: 1 - 10	Class
0	1000025	5	1	1	1	2	1	3	1	1	0
1	1002945	5	4	4	5	7	10	3	2	1	0
2	1015425	3	1	1	1	2	2	3	1	1	0
3	1016277	6	8	8	1	3	4	3	7	1	0
4	1017023	4	1	1	3	2	1	3	1	1	0
...	...	...	...	...	...	...	...	...	...	...	...
694	776715	3	1	1	1	3	2	1	1	1	0
695	841769	2	1	1	1	2	1	1	1	1	0
696	888820	5	10	10	3	7	3	8	10	2	1
697	897471	4	8	6	4	3	4	10	6	1	1
698	897471	4	8	8	5	4	5	10	4	1	1

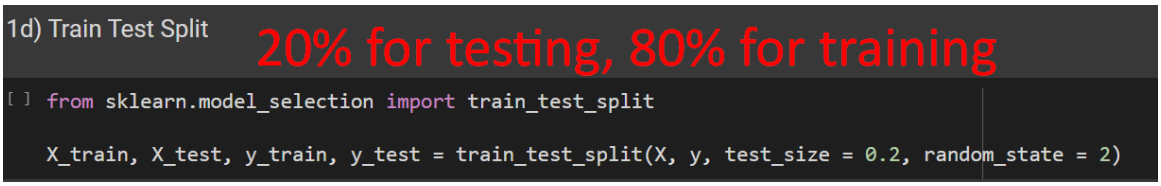
699 rows × 11 columns

C. WRANGLING THE X AND Y

Here we define our "Predictors (X)" and "Target (y)" by slicing out the columns



D. TRAIN TEST SPLIT



E. ANOTHER WAY TO GET THE DATA

1e) Another way to get the data....

```
[ ]  1 #Wget is a command-line tool that makes it possible to download files and interact with REST APIs.
     2
     3 !wget https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data
```

```
--2023-03-31 06:34:32--  https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data
Resolving archive.ics.uci.edu (archive.ics.uci.edu)... 128.195.10.252
Connecting to archive.ics.uci.edu (archive.ics.uci.edu)|128.195.10.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 19889 (19K) [application/x-httpd-php]
Saving to: 'breast-cancer-wisconsin.data'

breast-cancer-wisco 100%[===================>]  19.42K  --.-KB/s    in 0.1s

2023-03-31 06:34:32 (151 KB/s) - 'breast-cancer-wisconsin.data' saved [19889/19889]
```

```
1 #The 'ls' command is used to list files and directories
2 !ls
```

```
breast-cancer-wisconsin.data  sample_data
```

```
[ ]  1 !head breast-cancer-wisconsin.data
```

```
1000025,5,1,1,1,2,1,3,1,1,2
1002945,5,4,4,5,7,10,3,2,1,2
1015425,3,1,1,1,2,2,3,1,1,2
1016277,6,8,8,1,3,4,3,7,1,2
1017023,4,1,1,3,2,1,3,1,1,2
1017122,8,10,10,8,7,10,9,7,1,4
1018099,1,1,1,1,2,10,3,1,1,2
1018561,2,1,2,1,2,1,3,1,1,2
1033078,2,1,1,1,2,1,1,1,5,2
1033078,4,2,1,1,2,1,2,1,1,2
```

```
[ ]  1 import pandas as pd
     2
     3 df = pd.read_csv("breast-cancer-wisconsin.data", header = None)
```

```
[ ]  1 df.columns = ["ID",
     2               "Clump Thickness: 1 - 10",
     3               "Uniformity of Cell Size: 1 - 10",
     4               "Uniformity of Cell Shape: 1 - 10",
     5               "Marginal Adhesion: 1 - 10",
     6               "Single Epithelial Cell Size: 1 - 10",
     7               "Bare Nuclei: 1 - 10",
     8               "Bland Chromatin: 1 - 10",
     9               "Normal Nucleoli: 1 - 10",
    10               "Mitoses: 1 - 10",
    11               "Class: (2 for benign, 4 for malignant)"]
```

```
[ ]  1 df
```

	ID	Clump Thickness: 1 - 10	Uniformity of Cell Size: 1 - 10	Uniformity of Cell Shape: 1 - 10	Marginal Adhesion: 1 - 10	Single Epithelial Cell Size: 1 - 10	Bare Nuclei: 1 - 10	Bland Chromatin: 1 - 10	Normal Nucleoli: 1 - 10	Mitoses: 1 - 10	Class: (2 for benign, 4 for malignant)
0	1000025	5	1	1	1	2	1	3	1	1	2
1	1002945	5	4	4	5	7	10	3	2	1	2
2	1015425	3	1	1	1	2	2	3	1	1	2
3	1016277	6	8	8	1	3	4	3	7	1	2
4	1017023	4	1	1	3	2	1	3	1	1	2
...	...	...	...	...	...	...	...	...	...	...	...
694	776715	3	1	1	1	3	2	1	1	1	2
695	841769	2	1	1	1	2	1	1	1	1	2
696	888820	5	10	10	3	7	3	8	10	2	4
697	897471	4	8	6	4	3	4	10	6	1	4
698	897471	4	8	8	5	4	5	10	4	1	4

699 rows × 11 columns

F. ANOTHER WAY TO GET THE DATA USING TF.KERAS

1f) Another Way to Get the Files Using tf.keras....

```
[16] 1 url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/breast-cancer-wisconsin.data'
```

```
[17] 1 #install Tensorflow
     2
     3 !pip install -q tensorflow
     4 import tensorflow as tf
     5 print(tf.__version__)
```

```
2.12.0
```

```
[18] 1 tf.keras.utils.get_file('breast-cancer-wisconsin.data', url)
```

```
'/root/.keras/datasets/breast-cancer-wisconsin.data'
```

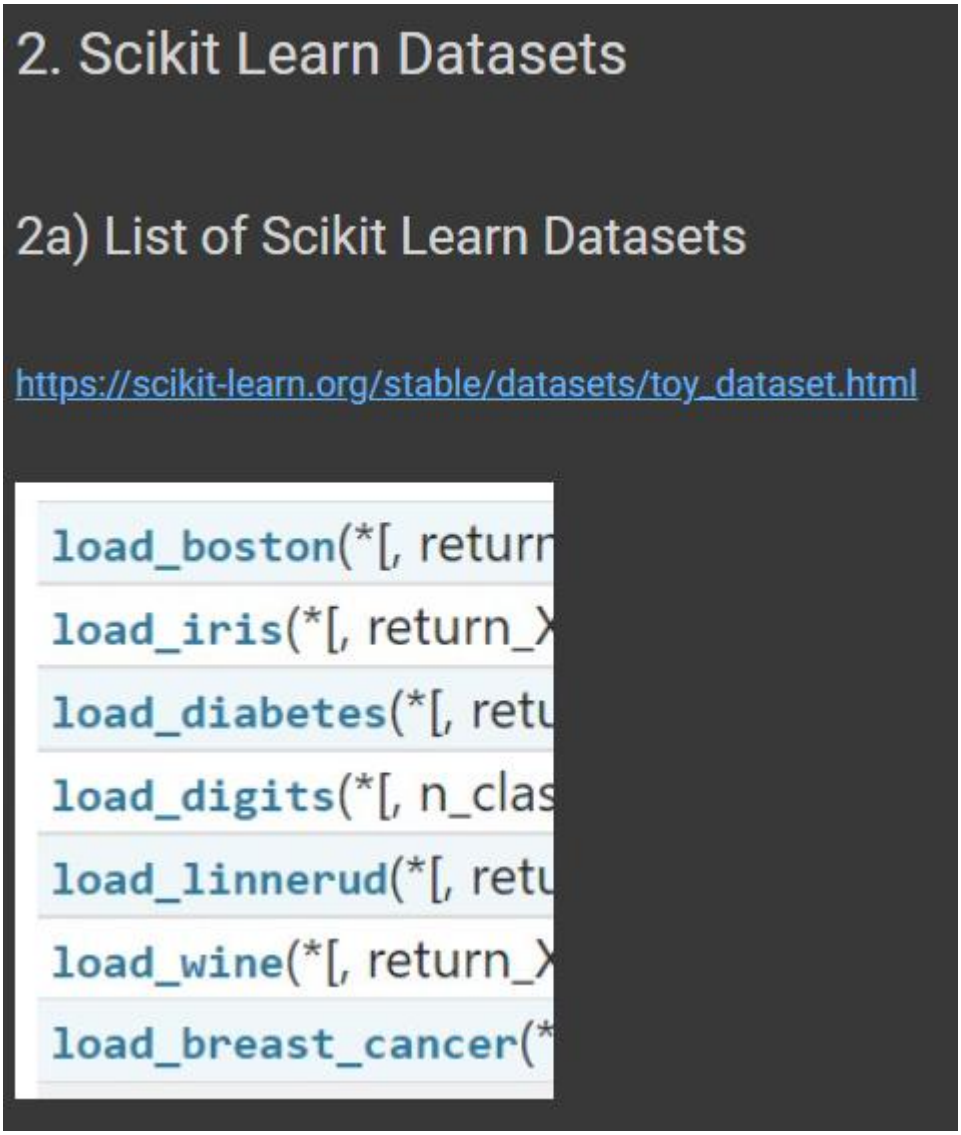
```
1 !head /root/.keras/datasets/breast-cancer-wisconsin.data
```

```
1000025,5,1,1,1,2,1,3,1,1,2
1002945,5,4,4,5,7,10,3,2,1,2
1015425,3,1,1,1,2,2,3,1,1,2
1016277,6,8,8,1,3,4,3,7,1,2
1017023,4,1,1,3,2,1,3,1,1,2
1017122,8,10,10,8,7,10,9,7,1,4
1018099,1,1,1,1,2,10,3,1,1,2
1018561,2,1,2,1,2,1,3,1,1,2
1033078,2,1,1,1,2,1,1,1,5,2
1033078,4,2,1,1,2,1,2,1,1,2
```

```
[20] 1 import pandas as pd
     2 df = pd.read_csv('/root/.keras/datasets/breast-cancer-wisconsin.data', header = None)
     3 df.head()
```

	0	1	2	3	4	5	6	7	8	9	10
0	1000025	5	1	1	1	2	1	3	1	1	2
1	1002945	5	4	4	5	7	10	3	2	1	2
2	1015425	3	1	1	1	2	2	3	1	1	2
3	1016277	6	8	8	1	3	4	3	7	1	2
4	1017023	4	1	1	3	2	1	3	1	1	2

A. LIST OF SCIKIT LEARN DATASETS



https://scikit-learn.org/stable/datasets/toy_dataset.html

B. GET IRIS DATASET

2b) Get IRIS Dataset

```
[ ] from sklearn.datasets import load_iris
    import pandas as pd

    data = load_iris()
    X = pd.DataFrame(data=data.data, columns=data.feature_names)
    X.head()

    #simply change the load_iris to load_wine to change the dataset
    #there are only 7 datasets above (picture) to choose from
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

```
y = pd.DataFrame(data = data.target, columns = ["Iris Class"])
y.sample(5)
```

	Iris Class
54	1
84	1
8	0
68	1
28	0

```
z = y["Iris Class"].map({0:'setosa', 1: 'versicolor', 2: 'virginica'})

#0: setosa
#1: versicolor
#2: virginica

print(z)

#just creating another column 'z' that does
#mapping of the Iris Class to Iris Type
```

```
0         setosa
1         setosa
2         setosa
3         setosa
4         setosa
        ...
145    virginica
146    virginica
147    virginica
148    virginica
149    virginica
Name: Iris Class, Length: 150, dtype: object
```

C. TRAIN TEST SPLIT

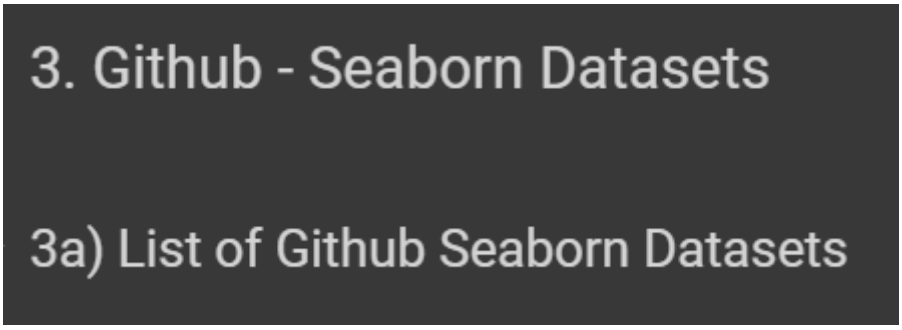
2c) Train Test Split

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 2)
```

IV. GITHUB – SEABORN DATASETS

A. LIST OF GITHUB SEABORN DATASETS



https://github.com/mwaskom/seaborn-data

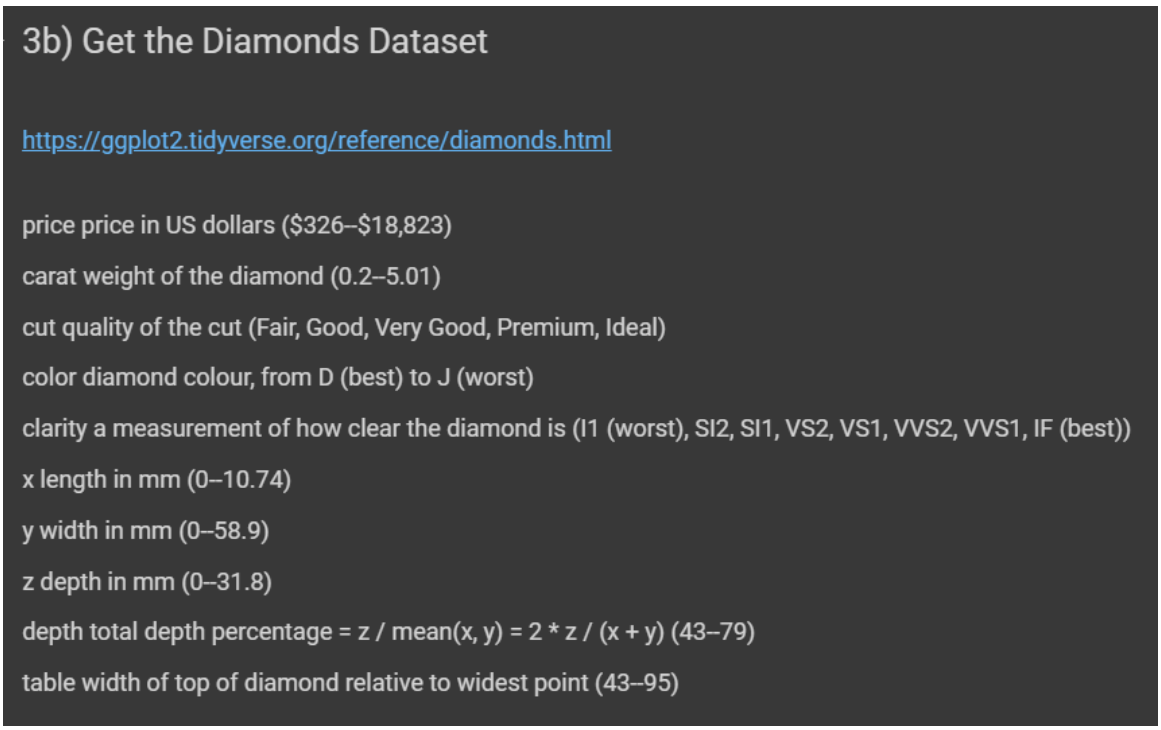
anagrams.csv	Rename messy anagrams dataset
anscombe.csv	Add anscombe dataset
attention.csv	Add attention dataset
brain_networks.csv	Add brain networks dataset
car_crashes.csv	Add 538 car crash dataset
diamonds.csv	Add diamonds dataset
dots.csv	Add dots dataset
dowjones.csv	Add dowjones dataset
exercise.csv	Add exercise dataset
flights.csv	Add flights dataset
fmri.csv	Change sorting of events in fmri data
geyser.csv	Add geyser dataset
glue.csv	Add several new datasets
healthexp.csv	Remove one-off 2021 datapoint from healthexp dataset
iris.csv	Add iris dataset
mpg.csv	Add mpg dataset
penguins.csv	Change culmen to bill in penguins dataset
planets.csv	Add planets dataset
seaice.csv	Add several new datasets
taxis.csv	Add green taxis to the taxis dataset
tips.csv	Add tips dataset
titanic.csv	Update titanic datset to remove index variable

Data sources

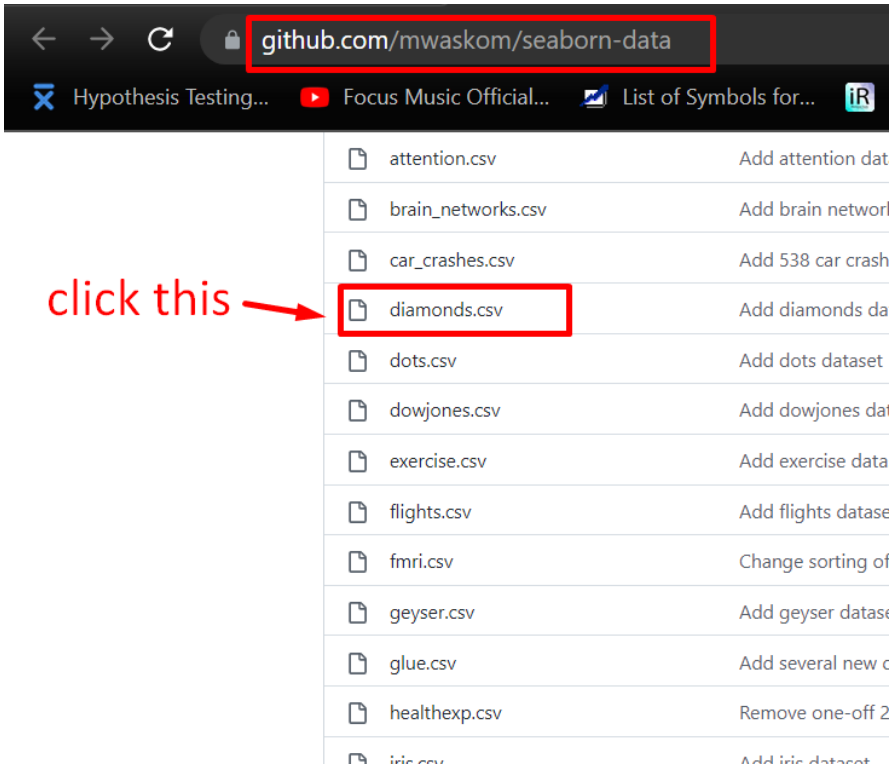
A partial list of where these datasets originate from.

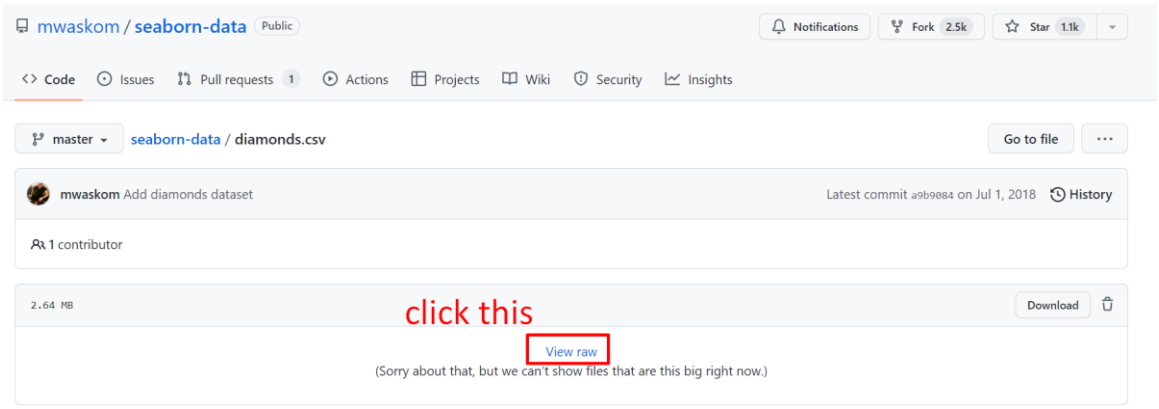
- `anagrams` : https://psych252.github.io/
- `anscombe` : https://en.wikipedia.org/wiki/Anscombe%27s_quartet
- `attention` : https://psych252.github.io/
- `car_crashes` : https://www.kaggle.com/fivethirtyeight/fivethirtyeight-bad-drivers-dataset
- `diamonds` : https://ggplot2.tidyverse.org/reference/diamonds.html
- `dots` : https://shadlenlab.columbia.edu/resources/RoitmanDataCode.html
- `dowjones` : https://fred.stlouisfed.org/series/M1109BUSM293NNBR
- `exercise` : https://psych252.github.io
- `fmri` : https://github.com/mwaskom/Waskom_CerebCortex_2017
- `geyser` : https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/faithful.html
- `glue` : https://gluebenchmark.com/leaderboard
- `healthexp` : https://ourworldindata.org/grapher/life-expectancy-vs-health-expenditure
- `iris` : https://archive.ics.uci.edu/ml/datasets/iris
- `mpg` : https://data.world/dataman-udit/cars-data
- `penguins` : https://github.com/allisonhorst/penguins
- `planets` : https://exoplanets.nasa.gov/exoplanet-catalog/
- `seaice` : https://nsidc.org/arcticseaicenews/sea-ice-tools/
- `taxis` : https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
- `tips` : https://rdrr.io/cran/reshape2/man/tips.html
- `titanic` : https://www.kaggle.com/c/titanic/data

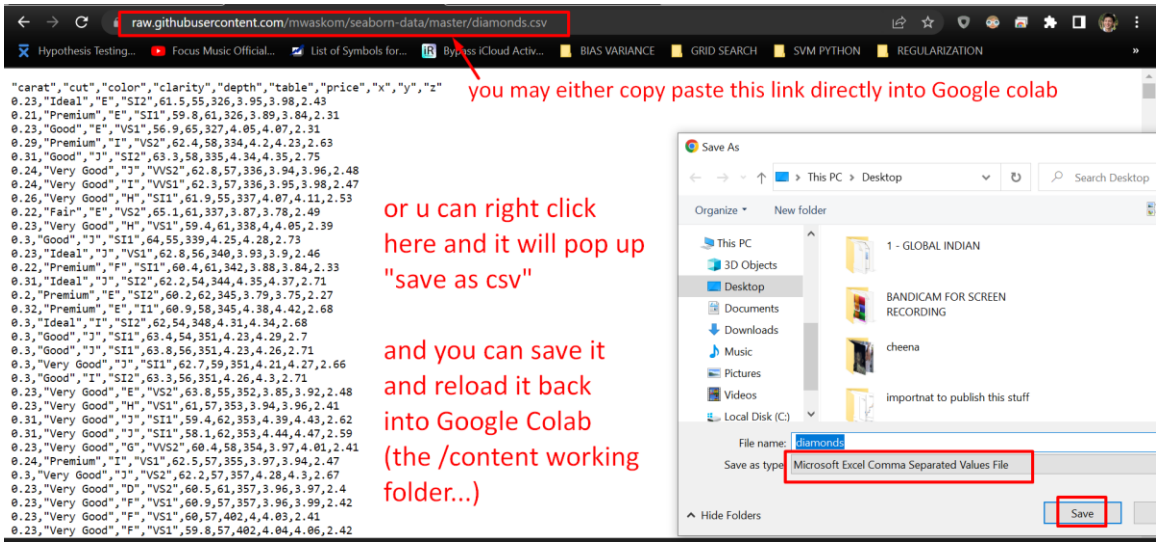
B. GET THE DIAMONDS DATASET



https://ggplot2.tidyverse.org/reference/diamonds.html








```
import pandas as pd

df = pd.read_csv('https://raw.githubusercontent.com/mwaskom/seaborn-data/master/diamonds.csv')

df
```

paste the link here

	carat	cut	color	clarity	depth	table	price	x	y	z
0	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
1	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
2	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
3	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
4	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
...	...	...	...	...	...	...	...	...	...	...
53935	0.72	Ideal	D	SI1	60.8	57.0	2757	5.75	5.76	3.50
53936	0.72	Good	D	SI1	63.1	55.0	2757	5.69	5.75	3.61
53937	0.70	Very Good	D	SI1	62.8	60.0	2757	5.66	5.68	3.56
53938	0.86	Premium	H	SI2	61.0	58.0	2757	6.15	6.12	3.74
53939	0.75	Ideal	D	SI2	62.2	55.0	2757	5.83	5.87	3.64

53940 rows × 10 columns

C. TRAIN TEST SPLIT

3c) Train Test Split

```
[ ] X_train = df.sample(frac=0.7, random_state = 0)
    y_train = X_train.pop('price')

    #70% for training, 30% for testing
```

```
[ ] X_test = df.drop(X_train.index)
    y_test = X_test.pop('price')
```

https://keras.io/api/datasets/

Available datasets

MNIST digits classification dataset

- load_data function

CIFAR10 small images classification dataset

- load_data function

CIFAR100 small images classification dataset

- load_data function

IMDB movie review sentiment classification dataset

- load_data function
- get_word_index function

Reuters newswire classification dataset

- load_data function
- get_word_index function

Fashion MNIST dataset, an alternative to MNIST

- load_data function

Boston Housing price regression dataset

- load_data function

A. MNIST HANDWRITTEN DIGITS

4a) MNIST Handwritten Digits

```
[ ] import tensorflow as tf
    from tensorflow import keras

    mnist = keras.datasets.mnist

    (X_train, y_train), (X_test, y_test) = mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
11493376/11490434 [==============================] - 0s 0us/step
11501568/11490434 [==============================] - 0s 0us/step
```

B. FASHION MNIST

4b) Fashion MNIST

```
[ ] import tensorflow as tf
    from tensorflow import keras

    fashion_mnist = keras.datasets.fashion_mnist
    (X_train, y_train), (X_test, y_test) = fashion_mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-labels-idx1-ubyte.gz
32768/29515 [=============================] - 0s 0us/step
40960/29515 [=========================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-images-idx3-ubyte.gz
26427392/26421880 [==============================] - 0s 0us/step
26435584/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-labels-idx1-ubyte.gz
16384/5148 [================================================================================================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-images-idx3-ubyte.gz
4423680/4422102 [==============================] - 0s 0us/step
4431872/4422102 [==============================] - 0s 0us/step
```

C. CIFAR 10

4c) CIFAR 10

```
[ ] import tensorflow as tf
    from tensorflow import keras

    cifar10 = keras.datasets.cifar10
    (X_train, y_train), (X_test, y_test) = cifar10.load_data()
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
170500096/170498071 [==============================] - 3s 0us/step
170508288/170498071 [==============================] - 3s 0us/step
```

A. GET DATASET LIST

5. Seaborn Datasets

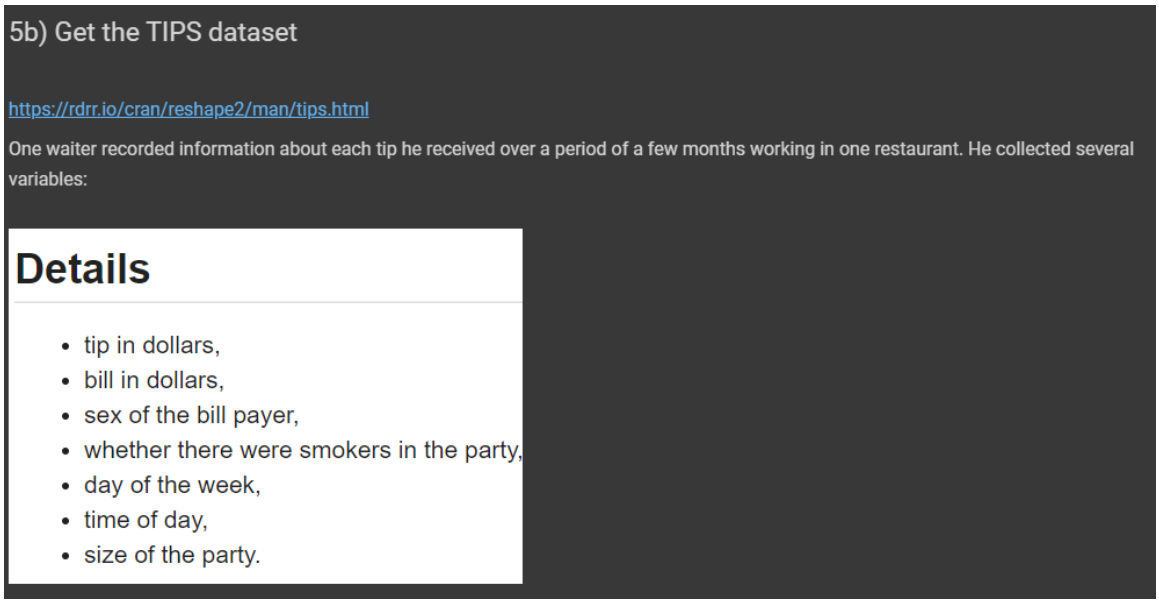
5a) Get Dataset List

```
import seaborn as sns

sns.get_dataset_names()
```

```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'dowjones',
 'exercise',
 'flights',
 'fmri',
 'geyser',
 'glue',
 'healthexp',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'seaice',
 'taxis',
 'tips',
 'titanic']
```

B. GET THE TIPS DATASET



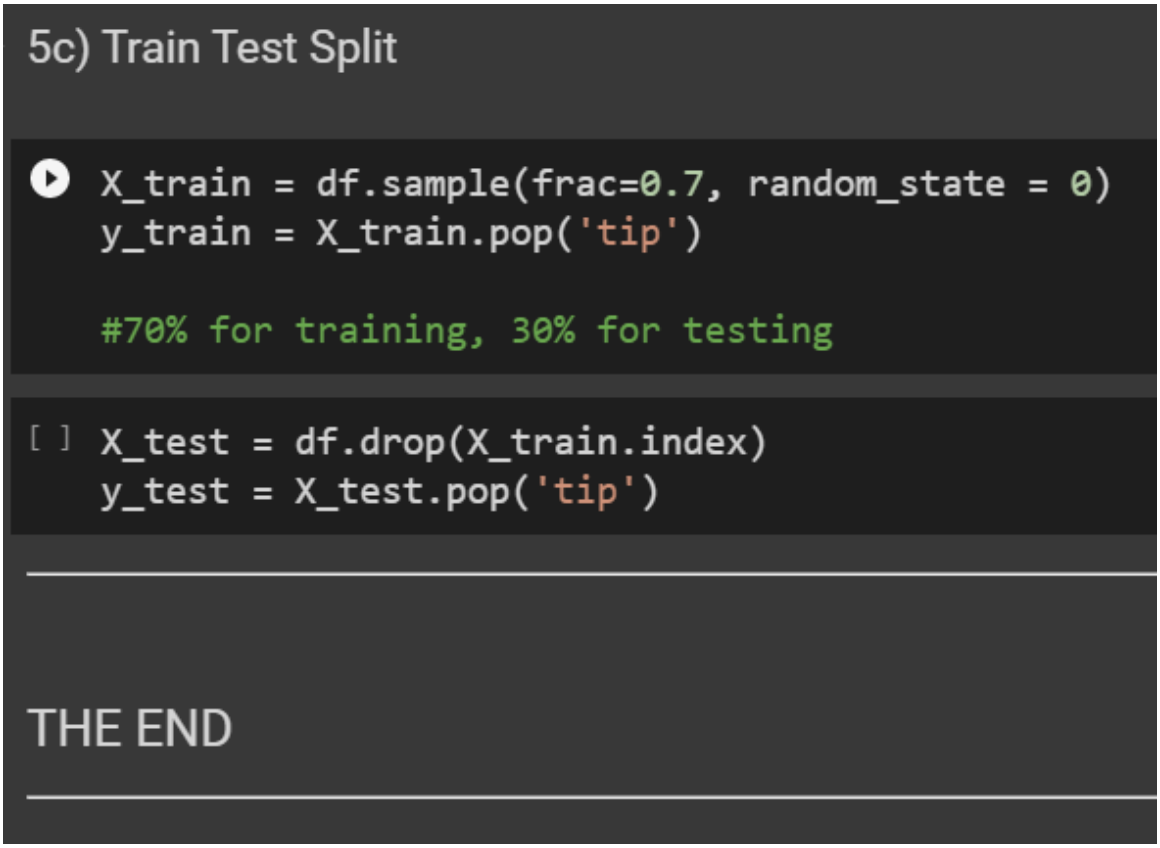
https://rdrr.io/cran/reshape2/man/tips.html

```
# Seaborn for plotting and styling

df = sns.load_dataset('tips')
df.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

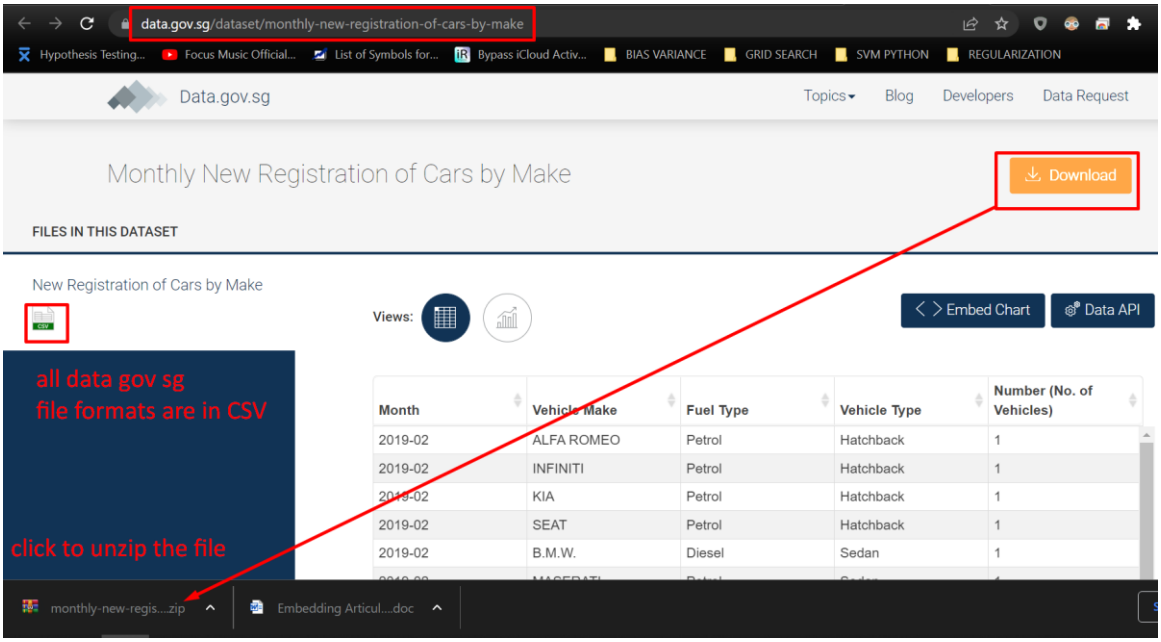
C. TRAIN TEST SPLIT

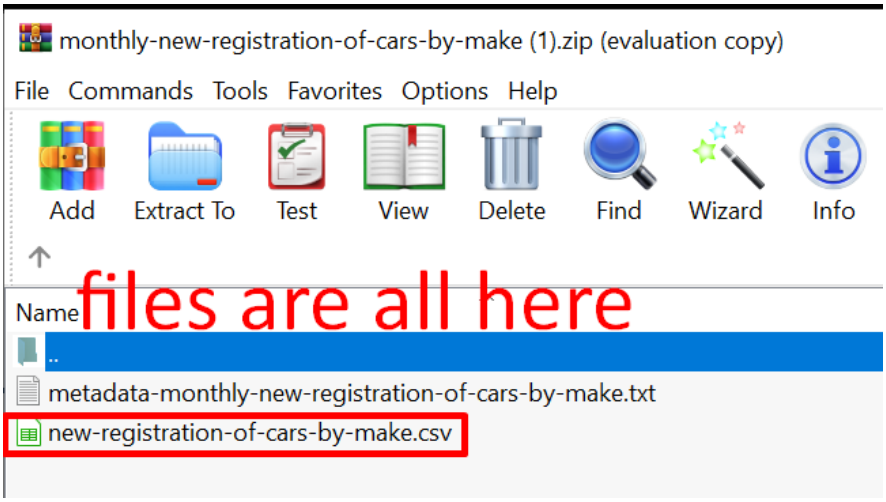


VII. DATA.GOV.SG

https://data.gov.sg/dataset/monthly-new-registration-of-cars-by-make

https://medium.com/@reiyasu/singapore-car-market-time-series-analysis-in-python-d254c9ec59c





ABOUT DR. ALVIN ANG

Dr. Alvin Ang earned his Ph.D., Masters and Bachelor degrees from NTU, Singapore. He is a scientist, entrepreneur, as well as a personal/business advisor. More about him at www.AlvinAng.sg.