**Assignment 1**

**Data Wrangling, I**

Perform the following operations using Python on any open source dataset (e.g., data.csv)

1. Import all the required Python Libraries.
2. Locate an open source data from the web ([e.g. https://www.kaggle.com](https://www.kaggle.com/)). Provide a clear description of the data and its source (i.e., URL of the web site).
3. Load the Dataset into pandas data frame.
4. Data Preprocessing: check for missing values in the data using pandas isnull(), describe() function to get some initial statistics. Provide variable descriptions. Types of variables etc. Check the dimensions of the data frame. and explain everything that you do to import/read/scrape the data set.

**What is Data Wrangling?**

Data wrangling can be defined as the process of cleaning, organizing, and transforming raw data into the desired format for analysts to use for prompt decision-making.

## Importance of Data Wrangling :

data professionals spend almost 80% of their time wrangling the data, leaving a mere 20% for exploration and modeling?

* Making raw data usable. Accurately wrangled data guarantees that quality data is entered into the downstream analysis.
* Getting all data from various sources into a centralized location so it can be used.
* Piecing together raw data according to the required format and understanding the business context of data
* Automated data integration tools are used as data wrangling techniques that clean and convert source data into a standard format that can be used repeatedly according to end requirements.
* Cleansing the data from the noise or flawed, missing elements
* Data wrangling acts as a preparation stage for the [data mining process](https://www.simplilearn.com/what-is-data-mining-article), which involves gathering data and making sense of it.
* Helping business users make concrete, timely decisions

**Data wrangling software performs six iterative steps**

1. Discovering
2. Structuring
3. Cleaning
4. Enriching
5. Validating
6. Publishing data

**Some examples of basic data wrangling tools are:**

* **Spreadsheets / Excel Power Query** - It is the most basic manual data wrangling tool
* **OpenRefine** - An automated data cleaning tool that requires programming skills
* **Tabula** – It is a tool suited for all data types
* **Google DataPrep** – It is a data service that explores, cleans, and prepares data
* **Data wrangler** – It is a data cleaning and transforming tool

**1 Import module in Python :**

* Import in python is similar to #include header\_file in C/C++. Python modules can get access to code from another module by importing the file/function using import.

import math

print(math.pi)

**syntax :**

**import module\_name.member\_name**

2.

* **syntax :**

**from module\_name import \***

* from math import pi

In the above code module, math is not imported, rather just pi has been imported as a variable.

All the functions and constants can be imported using \*.

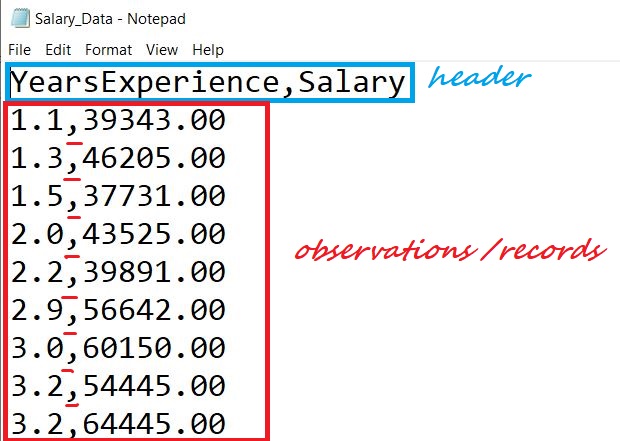
* **2 Open source data from web**
* [Open Data](https://en.wikipedia.org/wiki/Open_data) means the kind of data which is open for anyone and everyone for access, modification, reuse, and sharing.
* Following is the link to get open source dataset
* <https://rockcontent.com/blog/data-sources/>

you can download data and upload on colab for execution demonstration.

## What is a CSV?

“Comma Separated Values.” It is the simplest form of storing data in tabular form as plain text.

#### Structure of CSV:



## Reading a CSV

copy and paste this code in google colab and show output.

import csv

file = open("Salary\_Data.csv")

csvreader = csv.reader(file)

header = next(csvreader)

print(header)

rows = []

for row in csvreader:

rows.append(row)

print(rows)

file.close()

## 2 Implementing the above code using with() statement:

import csv

rows = []

with open("Salary\_Data.csv", 'r) as file:

csvreader = csv.reader(file)

header = next(csvreader)

for row in csvreader:

rows.append(row)

print(header)

print(rows)

### Using pandas:

**1. Import pandas library**

**2. Load CSV files to pandas using [read\_csv()](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html" \t "_blank)**

**Basic Syntax:**pandas.read\_csv(filename, delimiter=’,’)

**3. Extract the field names**

**.columns**is used to obtain the header/field names.

**4. Extract the rows**

All the data of a data frame can be accessed using the field names.

import pandas as pd

data= pd.read\_csv("Salary\_Data.csv")

data

data.columns

data.Salary

## 3. Writing to a CSV file

### 1 Using csv.writer

Let’s assume we are recording 3 Students data(Name, M1 Score, M2 Score)

header = ['Name', 'M1 Score', 'M2 Score']

data = [['Alex', 62, 80], ['Brad', 45, 56], ['Joey', 85, 98]]

filename = 'Students\_Data.csv'

with open(filename, 'w', newline="") as file:

csvwriter = csv.writer(file) # 2. create a csvwriter object

csvwriter.writerow(header) # 4. write the header

csvwriter.writerows(data) # 5. write the rest of the data

### 2 Using .writelines()

header = ['Name', 'M1 Score', 'M2 Score']

data = [['Alex', 62, 80], ['Brad', 45, 56], ['Joey', 85, 98]]

filename = 'Student\_scores.csv'

with open(filename, 'w') as file:

    for header in header:

        file.write(str(header)+', ')

    file.write('n')

    for row in data:

        for x in row:

            file.write(str(x)+', ')

        file.write('n')

### 3. Using pandas

header = ['Name', 'M1 Score', 'M2 Score']

data = [['Alex', 62, 80], ['Brad', 45, 56], ['Joey', 85, 98]]

data = pd.DataFrame(data, columns=header)

data.to\_csv('Stu\_data.csv', index=False)

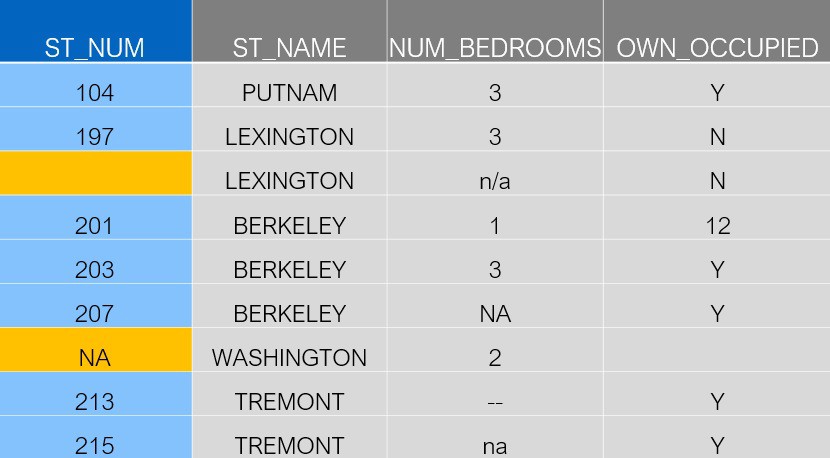
**Data cleaning** means fixing bad data in your data set.

Bad data could be:

* Empty cells
* Data in wrong format
* Wrong data
* Duplicates

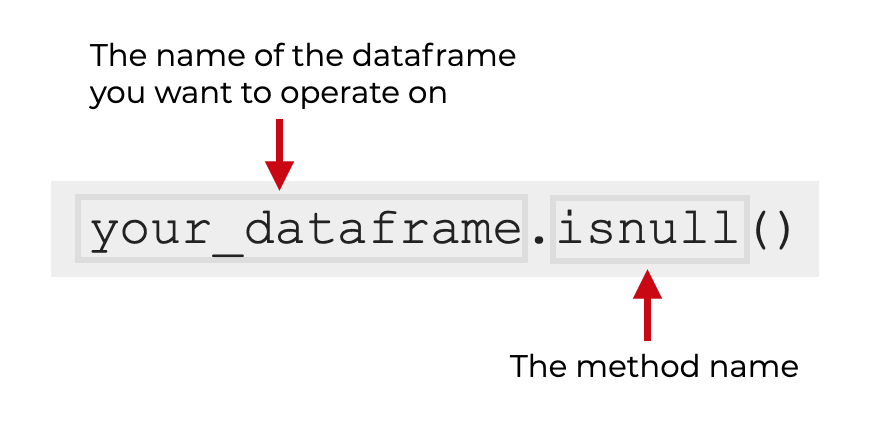
# Sources of Missing Values

* User forgot to fill in a field.
* Data was lost while transferring manually from a legacy database.
* There was a programming error.
* Users chose not to fill out a field tied to their beliefs about how the results would be used or interpreted.



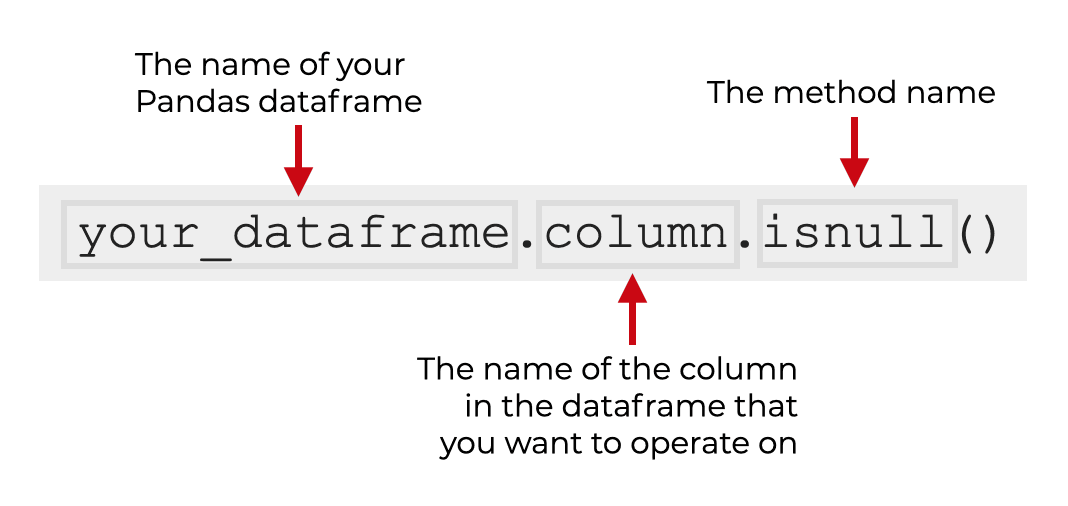
**isnull() on:**

* dataframes
* dataframe columns



The output will be an object of the same size as your dataframe that contains boolean True/False values. These boolean values indicate which dataframe values were missing.

### COLUMN SYNTAX



### COUNT THE MISSING VALUES IN EVERY COLUMN OF A DATAFRAME

(sales\_data

.isnull()

.sum()

)

OUT:

name 0

region 1

sales 2

expenses 2

dtype: int64

Empty cells can potentially give you a wrong result when you analyze data.

## 1.Remove Rows

import pandas as pd  
  
df = pd.read\_csv('data.csv')  
  
new\_df = df.dropna()  
  
print(new\_df.to\_string())

## 2. Replace Empty Values

Replace NULL values with the number 130:

import pandas as pd  
  
df = pd.read\_csv('data.csv')  
  
df.fillna(130, inplace = True)

### 3. Replace Only For Specified Columns

import pandas as pd  
  
df = pd.read\_csv('data.csv')  
  
df["Calories"].fillna(130, inplace = True)

## 4. Replace Using Mean, Median, or Mode

import pandas as pd  
  
df = pd.read\_csv('data.csv')  
  
x = df["Calories"].mean()  
  
df["Calories"].fillna(x, inplace = True)

# Pandas DataFrame describe() Method

The describe() method returns description of the data in the DataFrame.

If the DataFrame contains numerical data, the description contains these information for each column:

count - The number of not-empty values.  
mean - The average (mean) value.  
std - The standard deviation.  
min - the minimum value.  
25% - The 25% percentile\*.  
50% - The 50% percentile\*.  
75% - The 75% percentile\*.  
max - the maximum value.

## What are Pandas Dataframes

## lists and **numpy** arrays, store collections of values .

## **Python** lists are flexible and can store data items of various types (e.g. integers, floats, text strings),

## **numpy** arrays require all data elements to be of the same type.

A Pandas DataFrame is a 2 dimensional data structure, like a 2 dimensional array, or a table with rows and columns.

Each value in **pandas** dataframe is referred to as a cell that has a specific row index and column index within the tabular structure.

#load data into a DataFrame object:  
df = pd.DataFrame(data)

Print the data type of the variable x:

x = 5  
print(type(x))

## Setting the Specific Data Type

x = str("Hello World")

# Pandas DataFrame size Property

import pandas as pd  
  
df = pd.read\_csv('data.csv')  
  
print(df.size)

The size property returns the number of elements in the DataFrame.

The number of elements is the number of rows \* the number of columns.

So if u have 169 rows and 4 columns: 169 \* 4 = 676

## Viewing the Data

The head() method returns the headers and a specified number of rows, starting from the top.

import pandas as pd  
  
df = pd.read\_csv('data.csv')  
  
print(df.head(10))

if the number of rows is not specified, the head() method will return the top 5 rows.

There is also a tail() method for viewing the last rows of the DataFrame.

print(df.tail())

The DataFrames object has a method called info(), that gives you more information about the data set.

print(df.info())