Importing data

Data Science

What is a dataset?

- A dataset, or data set, is simply a collection of data.
- The simplest and most common format for datasets you'll find online is a spreadsheet or CSV format — a single file organized as a table of rows and columns. But some datasets will be stored in other formats, and they don't have to be just one file. Sometimes a dataset may be a zip file or folder containing multiple data tables with related data.

How are datasets created?

- Different datasets are created in different ways. In this post, you'll find links to sources with all kinds of datasets. Some of them will be machine-generated data. Some will be data that's been collected via surveys. Some may be data that's recorded from human observations. Some may be data that's been scraped from websites or pulled via APIs.
- Whenever you're working with a dataset, it's important to consider: how was this dataset created? Where does the data come from? Don't jump right into the analysis; take the time to first understand the data you are working with.

- A good place to find good data sets for data visualization projects are news sites that release their data publicly. They typically clean the data for you, and also already have charts they've made that you can replicate or improve.
- 1. FiveThirtyEight
- 2. BuzzFeed
- **3. NASA**
- **4. AWS Public Data sets :**
- **5. Kaggle**

Introduction

- Data scientists are expected to build high-performing machine learning models, but the starting point is getting the data into the Python environment. Only after importing the data can the data scientist clean, wrangle, visualize, and build predictive models on it.

CSV Files

- One of the most common data types is the *CSV* format, which is an acronym for *comma-separated values*. The general structure of CSV files uses rows as observations and columns as attributes.
- reads the .csv file and stores it as a pandas dataframe using the pandas pd.read_csv() function.

Text Files

- The other common flat file type is text files, which also contain textual data, but not necessarily in a tabular format.
- The first line of code below reads the text file using the pandas pd.read_table() function.
- data3 = pd.read_table("moby_dick.txt")
- print(data3)

Excel Data

- Excel data needs no introduction and is arguably the most widely used data type in the business world.
- imports and stores the dataset as a pandas dataframe, using the pandas pd.ExcelFile() function.

Importing Data from URL

- Often data is available on a website and can be downloaded into a local system.
- We will use the urllib library for performing this task, as this package provides the interface for fetching data across the web.
- urlretrieve function to save the file in the local environment.

SQL Database

- Relational databases are a prominent source of data storage for many organizations,
- import data from tables stored in SQL Server by building a connection.
- The pyodbc package is used in the illustration below. The next step is to establish the connection with the database for which you will need to have server, user, and database details to establish a connection

MySQL

- MySQL is currently the most popular database management system software used for managing the relational database.
- It is open-source database software, which is supported by Oracle Company.
- It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database
- Python needs a MySQL driver to access the MySQL database.
- We recommend that you use PIP to install "MySQL Connector".

Pandas - DataFrame - Loading the dataset from various data sources

- A dataset can be loaded from various data sources using relevant Pandas constructs (functions) as mentioned below:
- CSV file - read_csv() function
- JSON file - read_json() function
- Excel file - read_excel() function
- Database table - read_sql() function

Python code demonstrate creating DataFrame from dict narray / lists

- import pandas as pd   # Import pandas package
- # Define a dictionary containing employee data
- data = {'Name':['Jai', 'Princi', 'Gaurav', 'Anuj'],
-         'Age':[27, 24, 22, 32],
-         'Address':['Delhi', 'Kanpur', 'Allahabad', 'Kannauj'],
-         'Qualification':['Msc', 'MA', 'MCA', 'Phd']}
- # Convert the dictionary into DataFrame
- df = pd.DataFrame(data)
- print(df[['Name', 'Qualification']]) # select two columns

Accessing specific colums from dataframe

- # making data frame from csv file
- data = pd.read_csv("nba.csv", index_col ="Name")
- # retrieving row by loc method
- first = data.loc["Avery Bradley"]
- second = data.loc["R.J. Hunter"]
- print(first, "\n\n\n", second)

- # Read CSV file into a DataFrame
  df_csv = pd.read_csv(‘file.csv’)
- # Read Excel file into a DataFrame
  df_excel = pd.read_excel(‘file.xlsx’, sheet_name=’Sheet1')
- # Read JSON file into a DataFrame
  df_json = pd.read_json(‘file.json’)
- # Create a database engine
  engine = create_engine(‘sqlite:///database.db’)
- # Read SQL query result into a DataFrame
  df_sql = pd.read_sql(‘SELECT * FROM table_name’, engine)
- # Read HDF5 file into a DataFrame
  df_hdf5 = pd.read_hdf(‘file.h5’, key=’data’)

- # Read HTML tables into a list of DataFrames
  dfs_html = pd.read_html(‘file.html’)
- # Read fixed-width file into a DataFrame
  df_fwf = pd.read_fwf(‘file.txt’, widths=col_widths, header=None)
- # Read data from clipboard into a DataFrame
  df_clipboard = pd.read_clipboard()
- # Read CSV from a URL into a DataFrame
  url = ‘https://example.com/data.csv'
  df_url = pd.read_csv(url)

Zip Archive containing CSV files:

- import pandas as pd
  import zipfile
- # Extract CSV file from a zip archive and read it into a DataFrame
  with zipfile.ZipFile('archive.zip', 'r') as z:
  file_name = z.namelist()[0] # Assuming the first file is a CSV
  with z.open(file_name) as f:
  df_zip = pd.read_csv(f)

# Read Excel file with multiple sheets into a dictionary of DataFrames

-

```
xls_file = pd.ExcelFile(‘file_multi_sheets.xlsx’)
sheet_to_df_map = {sheet_name:
xls_file.parse(sheet_name) for sheet_name in
xls_file.sheet_names}
```

Google Sheets:

- import pandas as pd
- import gspread # Authenticate and open Google Sheets
- gc = gspread.service_account(filename=’path/to/credentials.json’)
- worksheet = gc.open(‘Your Google Sheet’).sheet1
- # Read Google Sheet into a DataFrame
- df_google_sheets = pd.DataFrame(worksheet.get_all_records())
- Note: For Google Sheets, you’ll need to install the gspread library and authenticate using a JSON key file.