Module 15: Python Data Science

Part A – Data Extraction

- Data Extraction is a technique of extracting information from websites.
- It is also called Web Harvesting or Web Scraping.
- Two prominent ways of extracting data:
    - Through APIs
    - Through Web Scraping
- Requests: it allows to us to send HTTP requests and response.
    - get(): this method is used for sending request to get data from the url specified.

Steps for connecting to website and getting information:

- ✓ Step 1: import libraries
- ✓ Step 2: send requests. In some case API key would be needed.
- ✓ Step3: make json
- ✓ Step 4: find element
- ✓ Step 5: print / process the element

- In order to get the data from webserver we make a GET request to get the information. To get request we pass the url.
- Server returns the status codes for every request that is made to a web server.
    - 200 ➔ everything is ok and result has been returned (if any)
    - 400 ➔ server thinks u made a bad request.
    - 401 ➔ server things u are not authenticated.
    - 403 ➔ resource access is forbidden.
    - 404 ➔ resource not found on server.
- Most API servers will send their responses in JSON formats. JSON is way to encode lists and dictionaries that are easily machine readable. To get content of a response as python object we use .json method on the response.

Application Program Interface:

- They are used to retrieve data from remote websites.
- To use an API we need to make a request to a remote web server and retrieve the data we need.
- API is useful in following cases:
  - The data is changing quickly.
  - We want small piece of data from larger set of data.

Steps for downloading a file:

- ✓ Step 1: import requests library
- ✓ Step 2: Find the url of the file to be downloaded
- ✓ Step3: Send a HTTP request to server and save HTTP response in response object.
- ✓ Step 4: open a new file in binary mode.
- ✓ Step 5: write the contents of response to new file

Beautiful Soup:

- Beautiful Soup is a Python package for parsing HTML and XML documents.
- It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for web scraping.
- BeautifulSoup: it allows us to pull out information from a webpage.
  - BeautifulSoup() will convert resposnse text from webpage into soup object.
  - find(): it will help to locate element in the soup.
  - find('element_name', {'selector name eg id/class', : 'value of selector'}

Steps for Web Scraping:

- ✓ Step 1: import libraries
- ✓ Step 2: send requests
- ✓ Step3: make soup object
- ✓ Step 4: find element
- ✓ Step 5: print / process the element

Part B – Data Analytics:

- Data science or Data analytics is process of analyzing large set of data to get answers on questions related to that data set.
- Pandas is a data science module that makes data science easy and effective.
- Pandas is an open source, BSD-licensed library providing high-performance, easy to use data structures and data analysis tools for the Python Programming language.
- Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the DataFrame.
- DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables.

Task 1: Reading CSV file:

- We use **read_csv()** method to do this task.
- To read a few columns we use usecols=['column-name1', 'column-name2']
- To read a few rows we use nrows=number-of-rows

Task 2a: DataFrame Selection Operations:

- **shape:**
  - which returns a tuple indicating the rows and the columns.
- **head():**
  - this method would return the first five rows from the DataFrame.
- **head(n):**
  - this method would return the first n rows from the DataFrame.
- **tail():**
  - this method would return the last 5 rows from the DataFrame.
- **tail(n):**
  - this method would return the last n rows from the DataFrame.
- Slicing operation is possible using [:] or [: :]
- *== is used for specifying the records filtering condition.*
- *str can help to apply string functions to the data.*

Task 2b: DataFrame Aggregate Operations:

- **sum():**
  - is used to find sum of all the columns.
- **sum(axis=1):**
  - is used to find sum of the rows.
- **max():**
  - is used to find max element of all the columns
- **max(axis=1):**
  - is used to find max element row-wise.
- **sort_values(by='column-name'):**
  - is used to sort the values column wise in ascending order. For descending we write : ascending =0 or ascending = False

Task 3: Writing the DataFrame into csv file:

- **to_csv():**
  - is used to write data frames to csv file.

Part C- Data Visualization:

Matplotlib:

- It is a graph plotting library in Python originally developed by John D. Hunter.
- It is the most popular graphing and data visualization module for Python which help data scientist to visualize their data or present it to someone.
- It supports varieties of graphs like:
  - ➢ Line plot
  - ➢ Bar charts
  - ➢ Pie Charts
  - ➢ And many more.
- We can control every element in the graph generated.
- We can also read data from file, plot the graph and save high quality outputs in formats like png, pdf, etc.

Line plot:

A line chart or line graph is a type of chart which displays information as a series of data points called 'markers' connected by straight line segments.

- plot(x, y):
  - it will plot x list and y list
- show():
  - it will show the plot
- title(string):
  - it will display the string as title
- xlabel() and ylabel():
  - it is for giving labels to x and y axis respectively.
- xlim() and ylim():
  - it is ued for controlling the x and y limits respectively.
- plot() customization:
  - we can specify color, linestyle, linewidth, marker, markerfacecolor and markersize
- savefig():
  - it can be used for saving the plot as .png/.pdf
- legend() and grid():
  - they are used for showing legend() and grid() respectively.

Bar Chart:

A bar chart or bar graph is a chart or graph that represents data with rectangular bars with heights or length proportional to the values that they represent.

- bar(x, y):
  - will draw a bar of x and their corresponding y.
- title(string):
  - it will display the string as title
- xlabel() and ylabel():
  - it is for giving labels to x and y axis respectively.
- show():
  - it will show the bar.
- legend() and grid():
  - they are used for showing legend() and grid() respectively.

Pie Chart:

*A **pie chart** (or a **circle chart**) is a circular statistical graphic which is divided into slices to illustrate numerical proportion.*

- pie():
  - will show the pie numerical proportion with the respective labels.
- axis():
  - will show a proper pie
- pie customization:
  - radius, explode, shadow, startangle, color and autopct.