Practical No. 03

Aim: To study and implement various methods of Pandas Library.

S/W Required: Python 3.9, Jupyter Notebook

Theory:

Introduction to Pandas:

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Pandas is a newer package built on top of NumPy and provides an efficient implementation of a DataFrame. DataFrames are essentially multidimensional arrays with attached row and column labels, and often with heterogeneous types and/or missing data. As well as offering a convenient storage interface for labeled data, Pandas implements a few powerful data operations familiar to users of both database frameworks and spreadsheet programs.

Pandas provides two data structures for performing data related operations:

1. Series: A Pandas Series is a one-dimensional array of indexed data. It can be created from:
   a. List
   b. Array
   c. Dictionary etc.
2. Dataframe: The next fundamental structure in Pandas is the DataFrame. Like the Series object, the DataFrame can be thought of either as a generalization of a NumPy array, or as a specialization of a Python dictionary.

Methods in Pandas:

1. pd.read_csv(), pd.read_excel() - They are used to read a CSV or an excel file to a pandas DataFrame format
2. df.head() - it shows the first 5 rows of the DataFrame
3. df.columns - this will print out all the columns of the dataset.

4. df.drop() - used to drop some unnecessary columns
5. df.len() - provides length of the dataframe
6. df.iloc() - This function takes as a parameter the rows and column indices and gives you the subset of the DataFrame accordingly.
7. df.loc() - This function does almost the similar operation as .iloc() function. But here we can specify exactly which row index we want and also the name of the columns we want in our subset.
8. df.dtypes - To know the datatypes of all the variable.
9. df.insert() - it inserts a column in the specified position
10. df.[“ ”].cumsum() - It provides you with the cumulative sum.
11. df.sample() - use this function to get a certain number of data points or a certain fraction or data point
12. df.column_name.unique() - It is used to find out the unique values of a categorical column
13. df.column_name.nunique() - It is used to find out the number (count) unique values of a categorical column *(can be used as df.nunique() to get the count of unique items in all columns)*
14. df.replace() - replaces the value of column
15. df.rename() - renames the column
16. df[‘column_name’].fillna() - replaces the null values with some other value of your choice
17. df.groupby() - You can group the data as per a certain variable and find out useful information about those groups
18. df.count() - It provides you the number of data in the DataFrame in the specified direction.
19. df[‘column_name’].describe() - function that provides some basic statistical measures
20. nlargest and nsmallest - This gives you the dataset with n number of largest values or smallest values of a specified variable.

Conclusion:

Thus, we have studied Pandas library and implemented methods associated with it, with proper examples.