**ASSIGNMENT HELP**

**MANUAL**



SUBMITTED

TO

VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE

FOR THE SKILL AND COMPETENCY EVALUATION OF

**DATA SCIENCE & MACHINE LEARNING**

IN

**CSE AI DEPARTMENT**

BY

**Bhushan Mane [22210472]**

**Class: S.Y. BTech Division: A Batch: A2**

**Batch Teacher**

**Dr.ANURADHA YENKIKAR.**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Contents** | **Page No.** |
| **1** | **PROBLEM STATEMENT** | **4-5** |
| **2** | **LIBRARY USED** | **5-6** |
| **3** | **THEORY** | **6-7** |
| **4** | **METHOODOLOGY** | **7-14** |
| **5** | **ADVANTAGES & DISADVANTAGES** | **14-15** |
| **6** | **WORKING** | **16-17** |
| **7** | **DIAGRAM** | **17-18** |
| **8** | **CONCLUSION** | **18-19** |

**1.PROBLEM STATEMENT:**

Perform the following operations using R/Python on suitable data sets:

a) read data from different formats (like csv, xls)

b) indexing and selecting data, sort data,

c) describe attributes of data, checking data types of each column,

d) counting unique values of data, format of each column, converting variable data type (e.g.

from long to short, vice versa),

e) identifying missing values and fill in the missing values

**2. LIBRARY USED:**

* pandas: For data manipulation and analysis
* xlrd: For reading Excel files

**3. THEORY:**

a) **Reading data from different formats**:

Reading data from diverse formats like CSV and XLS is pivotal in data analysis. CSV, being a plain text format, is widely used for its simplicity and compatibility across platforms. XLS, on the other hand, is preferred for its ability to store multiple sheets and complex formatting. Python libraries such as pandas offer dedicated functions like **read\_csv()** and **read\_excel()** to seamlessly import data from these formats into structured data frames, facilitating subsequent analysis and manipulation.

b) **Indexing and selecting data, sorting data**: Indexing and selecting data, alongside sorting, form the backbone of data exploration and analysis. Whether it's accessing specific rows or columns based on labels or positions or arranging data in a specified order, these operations are fundamental. Utilizing functionalities like **.loc[]** and **.iloc[]** in pandas allows for precise selection, while **sort\_values()** aids in organizing data based on chosen criteria, enhancing interpretability and facilitating downstream analysis.

c) **Describing attributes of data, checking data types of each column**: Describing data attributes offers insightful summaries crucial for understanding dataset characteristics. This includes metrics like count, mean, and standard deviation, providing a snapshot of numerical data distribution. Simultaneously, checking data types ensures data integrity, verifying whether columns contain numerical, categorical, or temporal data. Pandas' **describe()** and **dtypes** functions are invaluable tools for performing these tasks efficiently, empowering users to gain quick insights into dataset structure and content.

d) **Counting unique values of data, format of each column, converting variable data types**: Counting unique values in a dataset aids in identifying data patterns and anomalies, critical for accurate analysis and decision-making. Understanding the format of each column, i.e., its data type, is essential for proper data manipulation and interpretation. Furthermore, converting variable data types may be necessary to align data with analytical requirements. Pandas' **value\_counts()** and **dtypes** functions, along with **astype()**, offer robust solutions for efficiently performing these tasks, ensuring data consistency and compatibility across analyses.

e) **Identifying missing values and filling in the missing values**: Missing data can significantly impact the reliability and validity of analyses, necessitating thorough identification and appropriate handling. Detecting missing values enables users to gauge the extent of data incompleteness, informing subsequent strategies for imputation or exclusion. Filling in missing values, whether through mean imputation, mode substitution, or other techniques, helps preserve data integrity and ensures continuity in analysis. Utilizing pandas' **isnull().sum()** and **fillna()** functions streamlines this process, empowering users to address missing data systematically and effectively, ultimately enhancing the robustness of analytical outcomes.

**4. METHODS:**

a) **Reading data from different formats**:

Use **pd.read\_csv()** for CSV files and **pd.read\_excel()** for Excel files to read data into pandas DataFrames efficiently.

b) **Indexing and selecting data, sorting data**:

Utilize **.loc[]** and **.iloc[]** for precise data selection, and **sort\_values()** for arranging data based on specific columns.

c) **Describing attributes of data, checking data types of each column**:

Employ **describe()** for numerical column summaries and **dtypes** for checking data types of each column swiftly.

d) **Counting unique values of data, format of each column, converting variable data types**:

Use **value\_counts()** to count unique values, **dtypes** to check column formats, and **astype()** to convert variable data types efficiently.

e) **Identifying missing values and filling in the missing values**:

Detect missing values with **isnull().sum()** and fill them using **fillna()** with specified values for seamless data handling.

**5. ADVANTAGES AND DISADVANTAGES:**

**Advantages:** Performing operations such as reading data from different formats, indexing, selecting, sorting, and analyzing data attributes offers several advantages. These tasks are fundamental steps in data preprocessing, enabling data scientists and analysts to prepare and clean datasets efficiently. Reading data from diverse formats allows for the integration of information from various sources, facilitating comprehensive analysis. Additionally, indexing, selecting, and sorting data enable researchers to extract relevant subsets of data for in-depth examination, enhancing the accuracy and relevance of analytical outcomes. Moreover, analyzing data attributes and checking data types provides valuable insights into the structure and content of the dataset, aiding in identifying patterns, trends, and potential anomalies.

**Disadvantages:** Despite their advantages, these operations also present certain limitations and challenges. For instance, while reading data from different formats enhances flexibility, it requires expertise in handling various file types and may encounter compatibility issues. Similarly, while indexing and selecting data facilitate precise data manipulation, they can lead to increased memory usage and computational overhead, particularly for large datasets. Furthermore, while analyzing data attributes and converting data types are essential for data understanding and preparation, they may introduce errors or loss of information, especially if not performed carefully. Additionally, the process of identifying and handling missing values may introduce bias or inaccuracies in the analysis if not addressed appropriately, potentially impacting the reliability of results. Therefore, while these operations offer significant benefits, it is essential to be mindful of their limitations and employ best practices to mitigate potential drawbacks effectively.

Top of Form

**6. WORKING :**

The assignment involves executing fundamental data preprocessing tasks using Python's pandas library. Initially, data is imported from various file formats such as CSV and Excel into pandas DataFrames using **pd.read\_csv()** and **pd.read\_excel()**. Subsequently, operations like indexing, selecting, and sorting data are performed using **.loc[]**, **.iloc[]**, and **sort\_values()** respectively, enabling precise data manipulation and organization. Further analysis involves describing data attributes using **describe()** to obtain summary statistics and checking column data types with **dtypes**. Additionally, unique value counts are obtained using **value\_counts()**, while data type conversions are executed using **astype()**. Lastly, missing values are identified via **isnull().sum()** and filled using **fillna()**, ensuring data completeness. These tasks collectively lay the groundwork for data exploration and analysis, allowing for efficient data preparation and cleaning before more advanced analytical procedures.

**7. CONCLUSION:**

In conclusion, the comprehensive execution of fundamental data preprocessing tasks using Python's pandas library facilitates efficient data preparation and cleaning, laying a robust foundation for subsequent analytical procedures. By adeptly handling tasks such as reading data from diverse formats, indexing, selecting, sorting, and analyzing data attributes, alongside identifying and handling missing values, analysts can ensure data integrity, completeness, and accuracy. This systematic approach not only streamlines the data exploration and analysis process but also enhances the reliability and validity of insights derived from the dataset, ultimately empowering data-driven decision-making and fostering meaningful discoveries.

Top of Form