LAB. Manual Format for Data Science LAB.

(A sample experiment manual is given here. Write for other experiments similarly)

Experiment Number: 3

TITLE: Handling Missing Values in a Dataset

AIM: To perform various techniques to handle missing values in a dataset using Pandas Library.

THEORY:

Missing Values in dataset: Missing values in a dataset refer to the absence of data for one or more variables in a given observation.

They can occur for several reasons:

1. Data Collection Errors: Mistakes during data entry, measurement, or transmission can lead to missing data.
2. Non-response: In surveys or questionnaires, respondents may choose not to answer certain questions, leading to missing data.
3. Data Corruption: Data can be lost or corrupted during processing or storage due to hardware or software failures.
4. Intentional Omission: Sometimes, data might be intentionally left blank because it's irrelevant or unknown at the time of collection.

Handling missing values:

There are two ways to handle missing values:

- Removing missing values
- Imputing missing values

A. Removing missing values:

Remove Rows with Missing Values

- **Description**: This technique involves removing any row that contains at least one missing value.
- **Pros**: Simple and effective when the number of missing values is small.
- **Cons**: Can lead to significant data loss, especially if missing values are frequent.

Remove Columns with Missing Values

- **Description**: This method removes entire columns that contain any missing values.

- **Pros**: Useful if certain columns have a high percentage of missing values.
- **Cons**: Can lead to loss of potentially important features.

B. Imputing missing values:

a. Fill with a Specific Value

- **Description**: Missing values are replaced with a specific value, such as zero or a placeholder.
- **Pros**: Simple and quick.
- **Cons**: May introduce bias if the specific value does not represent the missing data accurately.

b. Fill with Mean, Median, or Mode

- **Description**: Missing values are replaced with the mean, median, or mode of the respective column.
- **Pros**: Preserves the overall distribution of the data.
- **Cons**: May not be suitable for skewed distributions or when the presence of missing values is not random.

c. Forward Fill and Backward Fill

- **Description**: Forward fill propagates the last valid observation forward, while backward fill propagates the next valid observation backward.
- **Pros**: Useful for time-series data where the last observation is a good estimate.
- **Cons**: Not suitable for all types of data, especially where trends change significantly over time.

d. Linear Interpolation

- **Description**: Estimates missing values using linear interpolation between neighboring points.
- **Pros**: Useful for time-series and numerical data.
- **Cons**: May not capture non-linear relationships effectively.

Pandas Function for handling missing values:

Here are various Pandas functions for handling missing values, along with their syntax and a brief description:

1. isnull()

- Syntax: df.isnull()

- Description: Returns a DataFrame of the same shape as the original, with `True` where values are missing and `False` where they are not.

2. notnull()

- Syntax: df.notnull()

- Description: Returns a DataFrame with `True` where values are not missing and `False` where they are missing.

3. dropna()

- Syntax: df.dropna(axis=0, how='any', inplace=False)

- Description: Removes missing values. By default, it drops rows where any value is missing. You can also drop columns (`axis=1`) or drop rows/columns where all values are missing (`how='all'`).

4. fillna()

- Syntax: df.fillna(value, method=None, axis=None, inplace=False)

- Description: Fills missing values with a specified value, method (`'ffill'` or `'bfill'`), or interpolates them.

5. replace()

- Syntax: df.replace(to_replace=np.nan, value=0, inplace=False)

- Description: Replaces missing values (or any specified values) with a specified value.

6. interpolate()

- Syntax: df.interpolate(method='linear', axis=0, inplace=False)

- Description: Interpolates missing values using a specified method such as linear interpolation.

7. mask()

- Syntax: df.mask(df.isnull(), other_value)

- Description: Replaces missing values with a given value, similar to `fillna()`.

8. ffill()

- Syntax: df.ffill(axis=0, inplace=False)

- Description: Propagates the last valid observation forward to fill missing values.

9. bfill()

- Syntax: df.bfill(axis=0, inplace=False)

- Description: Propagates the next valid observation backward to fill missing values.

10. dropna(subset=[])

- Syntax: df.dropna(subset=['column1', 'column2'], inplace=False)
- Description: Drops rows where specific columns have missing values.

Dataset Description:

The dataset of weather consists of data about weather of a city with columns such as day, temperature, windspeed, and events (like sunny, rainy, cloudy, snow). The missing values are shown using NaN.

Preview of a Dataset (Write first few records here):

	day	temperature	windspeed	event
0	2017-01-01	32.0	6.0	Rain
1	2017-01-04	NaN	7.0	Sunny
2	2017-01-05	28.0	NaN	Snow
3	2017-01-06	NaN	7.0	NaN
4	2017-01-07	32.0	NaN	Rain
5	2017-01-08	NaN	NaN	Sunny

Program (code using any Python Notebook):

(Note: Output should be embedded in the Python notebook itself)

Conclusion:

These pandas library functions provide flexibility in handling missing data, allowing you to clean and prepare datasets effectively.