Introduction to pandas Part 02

Working with missing Data

Reference: Python for Data Analysis - Data Wrangling...Pandas, NumPy, and IPython,2nd (Wes McKinney Oreilly 2017) - ***Chp07 Data Cleaning and Preparation***

More details to be given in the course: **IS674 Data Analysis, Visualization an d Use** (2nd Semester). Special tools exists for cleaning data such as OpenRefine (previously known as Google Refine) - https://openrefine.org/

Python's pandas library has functions to handle missing data (find, delete, or change missing data).

Example: The file: **`missing_values.csv`** contains some missing values in the columns `paused` and `volume`. Open the file in Excel and in the text editor to see the contents.

In **Excel** part of the file will look as follows:

user	video	playback position	paused	volume
cheryl	intro.html	5	FALSE	10
cheryl	intro.html	6		
cheryl	intro.html	9		
cheryl	intro.html	10		
bob	intro.html	1		
bob	intro.html	1		
bob	intro.html	1		
bob	intro.html	1		

In the **text editor** will look as follows:

```
user,video,playback position,paused,volume
cheryl,intro.html,5,FALSE,10
cheryl,intro.html,6,,
cheryl,intro.html,9,,
cheryl,intro.html,10,,
bob,intro.html,1,,
bob,intro.html,1,,
bob,intro.html,1,,
bob,intro.html,1,,
```

.....

When a dataframe for these data is created and the first five rows are displayed:

```
     user       video  playback position paused  volume
0  cheryl  intro.html                  5  False    10.0
1  cheryl  intro.html                  6    NaN     NaN
2  cheryl  intro.html                  9    NaN     NaN
3  cheryl  intro.html                 10    NaN     NaN
4     bob  intro.html                  1    NaN     NaN
```

Demo: In the notebook

Check for Missing Values

Use **`isna()`** or **`isnull()`** to detect values which are NaN

```
#Which elements are NaN in the DataFrame (True and False)
```

Demo: In the notebook

```
#Which elements are NOT NaN in the DataFrame (True and
False)
```

Use **`notna()`** or **`notnull()`** to check values which are **NOT** NaN

Demo: In the notebook

Note: The missing data are not taken care by **`describe()`** function.

More Examples on `isna()` and `notna()` are as follows:

Question: Which records have NaN in column 'paused'?
Demo: In the notebook

Question: Determine Number of NaN and NOT NaN in each **column**
Demo: In the notebook

Question: Determine Number of NaN and NOT NaN in each **row**
Demo: In the notebook

Question: Determine total number of NaN in the whole dataframe
Demo: In the notebook

#Qtn: In the demo above, why do we need double sum().sum()?

Using dataframe.count()

The `dataframe.count()` counts non-NaN cells for each column or row. That means does not include the NaN values.

Syntax: `DataFrame.count(axis=0, level=None, numeric_only=False)`

axis : {0 or 'index', 1 or 'columns'}, default 0

- If 0 or 'index' counts are generated for each column.
- If 1 or 'columns' counts are generated for each row.

Question: Determine Number of Non-NaN in Columns using count function
Demo: In the notebook

Question: Determine Number of Non-NaN in rows using count function
Demo: In the notebook

Deleting rows or columns with missing values

Use the function **dropna** which removes rows or columns with missing data.

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.dropna.html

Syntax:

```
DataFrame.dropna(axis=0, how='any', thresh=None, subset=None, inplace=False)
```

Parameters:

how : {'any', 'all'}, **default is 'any'**
Determine if row or column is removed from DataFrame, when we have at least one NA or all NA.

'any' : If any NA values are present, drop that row or column.
'all' : If all values are NA, drop that row or column.
'thresh' : int, optional. Require that many non-NA values.

inplace : bool, default False. If True, do operation inplace and return None.

Examples:

drop any row whose any element is equal to NaN

Demo: In the notebook

drop any column whose any element is equal to NaN

Demo: In the notebook

Note: Columns "paused" and "volume" are deleted.

drop all columns whose all elements are equal to NaN

Demo: In the notebook

Drop all columns where all element are missing

Demo: In the notebook

Note: No any column is deleted, since no any column contains all values as NaN

drop all rows whose all elements are equal to NaN

Demo: In the notebook

Dropping rows or columns using thresh parameter

thresh parameter can be considered to be threshold.

`thresh=N` requires that a row or column has **at least N non-NaNs** to survive. i.e. if the row or column has N non-null values or more, it should **NOT** be deleted.

Demo: In the notebook

Note: After dropping using thresh parameter, `volume` column is deleted.

Dropping rows using thresh parameter

Demo: In the notebook

Calculations with Missing Data

When summing data, NaN will be treated as Zero

Cleaning / Filling Missing Data

Pandas provides various methods for cleaning the missing values. The **`fillna`** function can "fill in" NaN values with non-null data in a couple of ways.

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.fillna.html

Syntax:

```
DataFrame.fillna(self, value=None, method=None, axis=None, inplace=False, limit=None, downcast=None, **kwargs
```

Parameters: Refer to a provided link above.

Examples:

Replace NaN in the dataframe with a Scalar Value

Demo: In the notebook

Replace NaN on specified columns with specified values

Demo: In the notebook

Replacing values in a dataframe with other values using replace function

Many times, a generic value is replaced with some specific value. This can be achieved by using **`replace`** method.

https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.replace.html

Syntax:

```
DataFrame.replace(self, to_replace=None, value=None, inplace=False, limit=None, regex=False, method='pad')
```

Demo: In the notebook

For more examples, refer to documentation in the given link

Limiting values outside the range (Outliers)

Data can be checked if they are within the allowable range. Example marks of students must be between 0 and 100 all inclusive. Use Boolean indexing.

Demo: In the notebook

String handling using str

https://pandas.pydata.org/docs/reference/api/pandas.Series.str.html

Pandas **Series** are equipped with a set of string processing methods that make it easy to operate on each element of the array. Perhaps most importantly, these methods exclude missing/NA values automatically. These are accessed via the **`str`** attribute and generally have names matching the equivalent (scalar) built-in string methods.

Demo: In the notebook

GroupBy

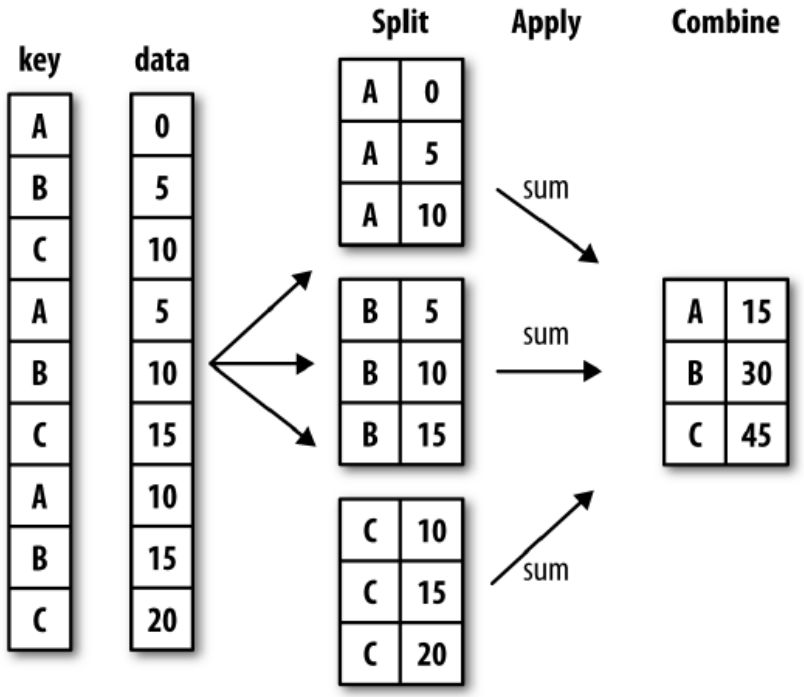
Groupby is one the most useful function available in pandas and it is worth you make sure you understand how to use it. Groupby does three things:

- It splits dataframe into groups based on some criteria
- It applies a **function** to each group independently
- Combine the results into a dataframe

https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.groupby.html

Syntax:

```
DataFrame.groupby(self, by=None, axis=0, level=None,
as_index=True, sort=True, group_keys=True, squeeze=False,
observed=False, **kwargs)
```



In above figure, sum function is used, but can be any other functions

Demo: In the notebook

groupby.agg

Aggregate using one or more operations over the specified axis.

https://pandas.pydata.org/docs/reference/api/pandas.core.groupby.GroupBy.agg.html

Syntax:

```
DataFrameGroupBy.agg(arg, *args, **kwargs)
```

agg is an alias for aggregate. Use the alias.

Demo: In the notebook (taken from above link)

Olympics Dataset Analysis

Dataset: `Olympics.csv`

Summer Olympic medallists 1896 to 2008. Use spreadsheet to view the contents, see why 4 rows are skipped using one of the options in reading csv function parameter **`skiprows.`**

Clarification of some Fields:

- Edition: Year
- NOC – Abbreviation of the country (National Olympic Committee)

The dataset has been cleaned in such a way there is no missing data (NaN)

Analysis of that dataset is found in the notebook.

=== *END* ===