Chapter 2.3 – Data Wrangling

Contents

- What is and Why the need for Data Wrangling
- Pandas and Dataframes
- Combining Dataframes
- Reshaping Dataframes
- Extracting Subsets
- Sorting Values and Index
- Handling Missing Values

Chapter 2.3 – Data Wrangling

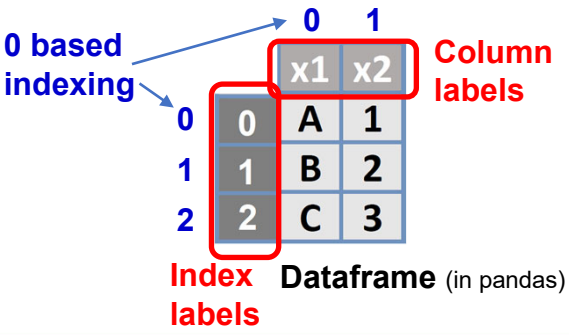
What is and Why the Need for Data Wrangling?

- Data wrangling is the process of **cleaning**, **structuring** and **enhancing** raw data into a format that is more appropriate and readily used for analysis & visualisation.
- The data available is often not in a **suitable form** for use with data visualisation tools. Some examples include:
  - Data needed are stored in **different files**. They need to be retrieved and combined into a single table for analysis.
  - Data available is extensive. Only a **subset is needed** for analysis.
  - Data in the table is **not organised** in the manner suitable for analysis.
  - Data in the table has **missing** or **outlier values** that can affect the analysis.
  - **New enhanced data** (e.g computed mean) must be entered in the table for analysis.

Data Wrangling - Getting Started

Pandas and Dataframes

- **Pandas** is a powerful and easy to use **open-source** data analysis and manipulation **library**, built on top of the **Python** programming language.
- **Dataframes** are used in pandas for storing data in rows (observations) and columns (variables or dimensions).
  - **Columns** are dimension of observations (variables) and are given a **label** (e.g. **x1**).
  - The row labels are referred to as **index** and has default labels of (0, 1, 2..) but can be renamed to any label.
  - Dataframes uses **0 based indexing**.



NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

[1] Pandas – Python Data Analysis Library - https://pandas.pydata.org/

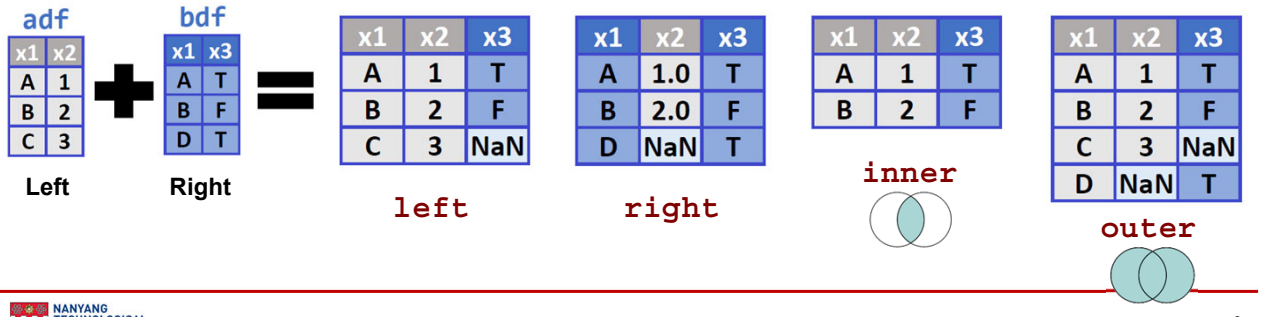
3

3

Combining Dataframes

Merge

- Data from two dataframes can be combined into a dataframe in many ways using the `merge` operator.
- Two dataframes can be **merged** using qualifiers like **left**, **right**, **inner** and **outer**.
- **Missing values** in the merged dataframe are given the **NaN** (Not a number) values.



NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

[2] For syntax of **merge** - see Data Wrangling with Pandas (Cheat Sheet) - https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf

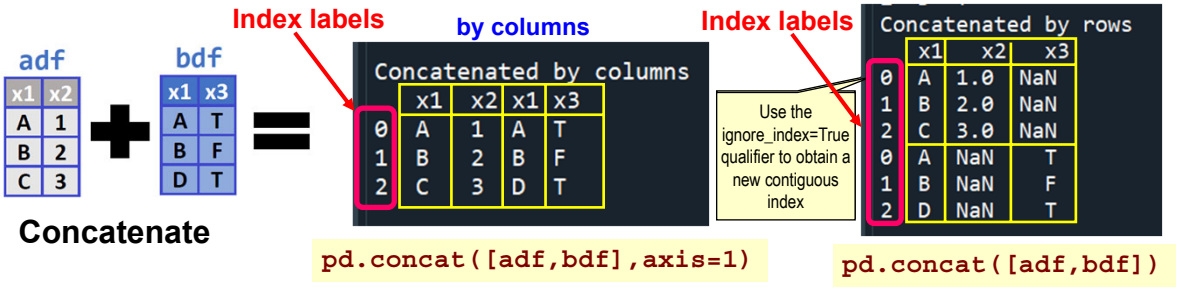
4

4

Combining Dataframes

Concatenate

- Data from 2 dataframes (or the same frame itself) can be appended using `concat`.
- The dataframes can be appended based on **rows** or **columns** (use `axis=1`)
- **Missing values** in the concatenated dataframe are given the **NaN** values.



[3] For syntax of **concat**- see pandas ref at - https://pandas.pydata.org/docs/reference/api/pandas.concat.html

5

5

Reshaping Dataframes

Pivot

- Data reshaping **rearranges** the form of the data **without changing** the content of the dataset so as to facilitate the next stage of analysis (e.g. plotting a graph).
- The `pivot` operator reshapes a table in the **long form to** a **wide form** that is more amenable to analysis as the desired variables are in columns.

Expenses

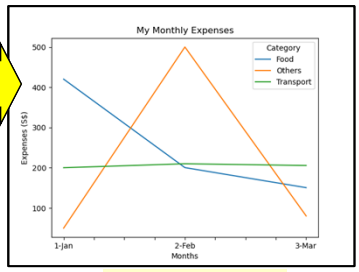
Month	Category	Amount
1-Jan	Food	420.30
1-Jan	Transport	200.20
1-Jan	Others	50.00
2-Feb	Food	200.45
2-Feb	Transport	210.00
2-Feb	Others	500.23
3-Mar	Food	150.60
3-Mar	Transport	205.70
3-Mar	Others	80.30

pivot

ExpPlot

Month	Food	Transport	Other
1-Jan	420.30	200.20	50.00
2-Feb	200.45	210.00	500.23
3-Mar	150.60	205.70	80.30

```
ExpPlot =
Expenses.pivot(index='Month',
               columns='Category',
               values='Amount')
```




```
ExpPlot.plot()
```

[4] For syntax of **pivot** - see pandas ref at - https://pandas.pydata.org/docs/user_guide/reshaping.html

6

6

Reshaping Dataframes

Melt

- The `melt` operator reshapes a table in the **wide form to** a **long form**.
- It creates a dataframe where one (or more) columns are **identifier variables**, while all other columns (considered measured variables), are unpivoted to the row axis to form a **variable** and a **value** column.

Temp

Name	Mon	Tue	Wed	Thu	Fri	Sat	Sun
John	36.5	36.8	36.4	35.8	36.0	36.5	35.8
Ahmad	36.5	36.6	36.7	36.8	36.6	36.7	36.8
Ginny	35.5	36.0	36.2	37.0	37.5	37.0	38.3
Manish	37.5	37.0	36.5	37.3	36.6	36.5	36.0

melt

Name	Variable	Value
John	Mon	36.5
Ahmad	Mon	36.5
Ginny	Mon	35.5
Manish	Mon	37.5
John	Tue	36.8
Ahmad	Tue	36.6
Ginny	Tue	36.0
Manish	Tue	37.0
John	Wed	36.4
Ahmad	Wed	36.7
Ginny	Wed	36.2
Manish	Wed	36.5
:	:	:
John	Sun	35.8
Ahmad	Sun	36.8
Ginny	Sun	38.3
Manish	Sun	36.0

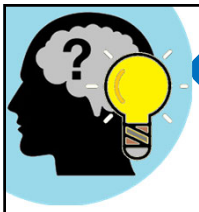
```
TempTable = Temp.melt(id_vars='Name')
```

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

[5] For syntax of **melt** - see pandas ref at - https://pandas.pydata.org/docs/reference/api/pandas.melt.html

7

7



Think and Apply

Reshaping Dataframe with Pivot and Melt

- The table in the dataset "*Daily Temperature.csv*" list the temperatures of four students in columns, according to each day of the week.
- Create a chart of daily temperatures of each student, each with their own line plot.

Temp

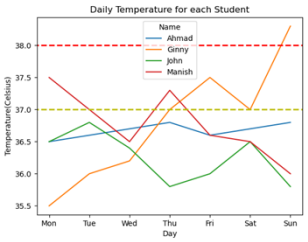
Name	Mon	Tue	Wed	Thu	Fri	Sat	Sun
John	36.5	36.8	36.4	35.8	36.0	36.5	35.8
Ahmad	36.5	36.6	36.7	36.8	36.6	36.7	36.8
Ginny	35.5	36.0	36.2	37.0	37.5	37.0	38.3
Manish	37.5	37.0	36.5	37.3	36.6	36.5	36.0

Reshape **Temp** so that each student's daily temperature in a column.

TempPlot

Name	Ahmad	Ginny	John	Manish
Day				
Mon	36.5	35.5	36.5	37.5
Tue	36.6	36.0	36.8	37.0
Wed	36.7	36.2	36.4	36.5
Thu	36.8	37.0	35.8	37.3
Fri	36.6	37.5	36.0	36.6
Sat	36.7	37.0	36.5	36.5
Sun	36.8	38.3	35.8	36.0

plot



Use **TempPlot.plot()** to create the line chart

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

8

8

Extracting Subsets

Data Slicing

- Most datasets have more data then is required for the current analysis. Data slicing refers to the process of extracting the **required subset of data** from the dataset.
- Data slicing in pandas can be done using operators `loc` and `iloc`.

Temp

Index	Name	Mon	Tue	Wed	Thu	Fri	Sat	Sun
0	John	36.5	36.8	36.4	35.8	36.0	36.5	35.8
1	Ahmad	36.5	36.6	36.7	36.8	36.6	36.7	36.8
2	Ginny	35.5	36.0	36.2	37.0	37.5	37.0	38.3
3	Manish	37.5	37.0	36.5	37.3	36.6	36.5	36.0

loc — Uses labels

	Name	Sat	Sun
0	John	36.5	35.8
1	Ahmad	36.7	36.8
2	Ginny	37.0	38.3
3	Manish	36.5	36.0

Students' temperature over the weekend

```
Temp.loc[:,['Name','Sat','Sun']]
```

iloc — Uses index number

Extract students with any temperature > 37°C

	Name	Mon	Tue	Wed	Thu	Fri	Sat	Sun
2	Ginny	35.5	36.0	36.2	37.0	37.5	37.0	38.3
3	Manish	37.5	37.0	36.5	37.3	36.6	36.5	36.0

```
Temp.loc[(Temp.iloc[:,1:]>37).any(axis=1)]
```

	Mon	Tue	Wed	Thu	Fri
0	36.5	36.8	36.4	35.8	36.0
3	37.5	37.0	36.5	37.3	36.6

Temperatures of 1st and 4th students in the weekdays

```
Temp.loc[[0,3],'Mon':'Fri']
```


[6] Pandas reference for data slicing- https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html

Rearranging Data

Sorting values and index

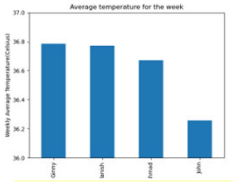
- Data in a table may need to be sorted based on one or more selected **attributes** to facilitate further processing (e.g. plotting an ascending bar chart)
- Pandas provides the `sort_values` and `sort_index` operators for this purpose.

Temp

Index	Name	Mon	Tue	Wed	Thu	Fri	Sat	Sun
0	John	36.5	36.8	36.4	35.8	36.0	36.5	35.8
1	Ahmad	36.5	36.6	36.7	36.8	36.6	36.7	36.8
2	Ginny	35.5	36.0	36.2	37.0	37.5	37.0	38.3
3	Manish	37.5	37.0	36.5	37.3	36.6	36.5	36.0

Get mean & sort

Ginny	36.785714
Manish	36.771429
Ahmad	36.671429
John	36.257143




```
BarPlot.plot.bar()
```

Get mean over numeric columns then sort in descending order

```
BarPlot = Temp.mean(axis=1)
BarPlot.sort_values(ascending=False)
```

sort

	Name	Mon	Tue	Wed	Thu	Fri	Sat	Sun
3	Manish	37.5	37.0	36.5	37.3	36.6	36.5	36.0
0	John	36.5	36.8	36.4	35.8	36.0	36.5	35.8
2	Ginny	35.5	36.0	36.2	37.0	37.5	37.0	38.3
1	Ahmad	36.5	36.6	36.7	36.8	36.6	36.7	36.8

Sort Name in descending order

```
Temp.sort_values(by='Name',ascending=False)
```

```
Temp.sort_index()
```

Sort index back to original order

[7] Ref to pandas sorting - https://realpython.com/pandas-sort-python/ & https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.sort_values.html

Handling Missing Values

Pandas operators for NaN values

- Raw datasets and reshaped dataframes often contain missing or **null values** (NaN). These null values need to be accounted for and replaced/removed before further processing or data analysis.
- Pandas provide various operators for **detecting** and **handling** null values.

df

x1	x2
A	1
B	NaN
C	3

Returns **True** for null values

	x1	x2
0	False	False
1	False	True
2	False	False

`df.isnull()`

Remove rows with null values

	x1	x2
0	A	1.0
2	C	3.0

`df.dropna()`

Remove col with null values

	x1
0	A
1	B
2	C

`df.dropna(axis=1)`

Fill null values with 0

	x1	x2
0	A	1.0
1	B	0.0
2	C	3.0

`df.fillna(0)`

Fill null values with mean of col

	x1	x2
0	A	1.0
1	B	2.0
2	C	3.0

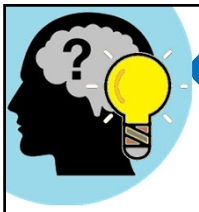
`df.fillna(df.mean())`

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

[8] Pandas reference for handling missing data - https://pandas.pydata.org/pandas-docs/stable/user_guide/missing_data.html

11

11



Think and Apply

Data Wrangling Exercise

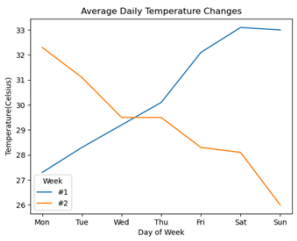
- Two temperature tables for week #1 and #2 are stored in **two different datafiles**.
- **Missing value** needs to be interpolated from its nearest data points.
- **Combine** and **reshape** the two dataframes to produce **two different line plots**. One shows the two weeks separately, the other shows the two as a single line.

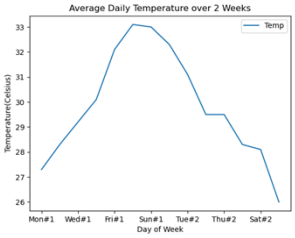
Day	Week	Temp
Mon	#1	27.3
Tue	#1	28.3
Wed	#1	29.2
Thu	#1	30.1
Fri	#1	32.1
Sat	#1	33.1
Sun	#1	33.0

Temp1

Day	Week	Temp
Mon	#2	32.3
Tue	#2	31.1
Wed	#2	29.5
Thu	#2	29.5
Fri	#2	28.3
Sat	#2	28.1
Sun	#2	26.0

Temp2





NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

12

12

Summary

Data Wrangling

- Data wrangling is an important step in the process of data visualisation as it **prepares** that data into a **format** that can be **readily plotted** with the various visualisation routines.
- Data created or loaded from various stored sources need to be **combined** into a suitable single data structure called a **dataframe**.
- Dataframes often need to be **reshaped** into the **appropriate format** so that it is suitable for used with different visualisation routines.
- Data in dataframes may also need to be **sorted** into a specific order for display.
- **Missing data** need to be **filled in** or **removed** in order to ensure they do not **interfere** with the visualisation process.

References – Data Attributes and Wrangling

[1] Pandas – Python Data Analysis Library - https://pandas.pydata.org/
[2] For syntax of merge - see Data Wrangling with Pandas (Cheat Sheet) - https://pandas.pydata.org/Pandas_Cheat_Sheet.pdf
[3] For syntax of concat- see pandas ref at - https://pandas.pydata.org/docs/reference/api/pandas.concat.html
[4] For syntax of pivot - see pandas ref at - https://pandas.pydata.org/docs/user_guide/reshaping.html
[5] For syntax of melt - see pandas ref at - https://pandas.pydata.org/docs/reference/api/pandas.melt.html
[6] Pandas reference for data slicing- https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
[7] Ref to pandas sorting - https://realpython.com/pandas-sort-python/ & https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.sort_values.html
[8] Pandas reference for handling missing data - https://pandas.pydata.org/pandas-docs/stable/user_guide/missing_data.html

Note: All online articles were accessed between May to June 2021