

NTI

Data Wrangling Report

Aisha Amr Hassan

Asmaa Adel

Esraa EL-Sayed

Abdelaziz Sakr

Supervised by

Eng.Youssef Ezz Eldeen



Introduction To Data Wrangling

Data wrangling, also called data cleaning, data remediation, or data munging, is the process of transforming raw data into a more usable format. This involves cleaning, structuring, and enriching the data so that it's ready for analysis.

Importance Of Data Wrangling

Data wrangling is crucial for ensuring the accuracy, consistency, and relevance of your dataset. It helps eliminate missing values, correct inconsistencies, and remove irrelevant information, transforming raw data into a high-quality, structured format.

A well-organized dataset is fundamental for:

- Building reliable machine learning models.
- Generating insightful visualizations.
- Performing accurate statistical analyses.

High-quality data reduces the risk of errors and biases, leading to more trustworthy insights and better-informed decision-making.

Types Of Data Wrangling

Data Wrangling has two types:

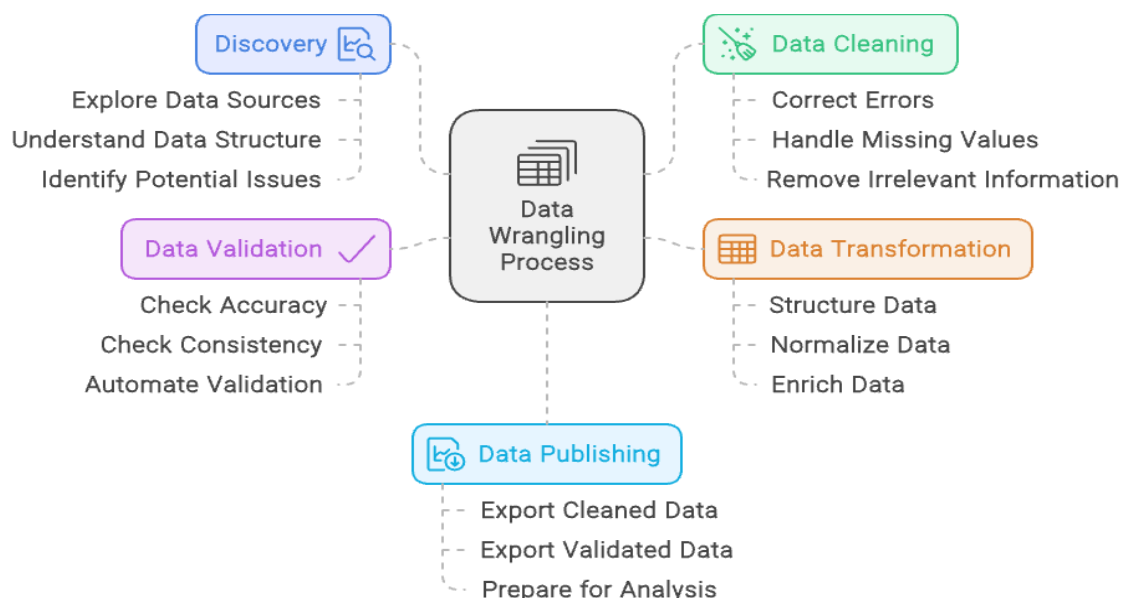
1. **Data Quality Issues:** These refer to problems or shortcomings in the qualitative or quantitative state of information, such as duplicate data, inaccurate and missing data, inconsistent data, irrelevant data, etc.

2. **Data Tidiness Issues:** These are related to the structure of the data, such as when a column header is a value, multiple variables are stored in one column, multiple columns represent one variable, variables are stored as rows, or observations are spread across multiple tables.

Steps Taken in Data Wrangling

There are 5 main steps to data wrangling:

1. **Discovery:** Explore and understand the data to identify its sources, structure, and potential issues.
2. **Data Cleaning:** Correct errors, handle missing values, and remove irrelevant information to ensure data quality.
3. **Data Transformation:** Structure, normalize, and enrich data to align with analysis needs.
4. **Data Validation:** Check the accuracy and consistency of the data with automation in order to ensure it's reliable.
5. **Data Publishing:** Export the cleaned and validated data in a format that's ready for analysis, often through dashboards and reports, or further use.



Dataset Used in the Project

The dataset used in this project is derived from a supermarket dataset, aiming to uncover insights into customer behavior, product performance, customer location, total sales, and ratings.

Dataset columns:

- Invoice id: Computer generated sales slip invoice identification number
- Branch: Branch of supercenter (3 branches are available identified by A, B and C).
- City: Location of supercenters
- Customer type: Type of customers, recorded by Members for customers using member card and Normal for without member card.
- Gender: Gender type of customer
- Product line: General item categorization groups - Electronic accessories, Fashion accessories, Food and beverages, Health and beauty, Home and lifestyle, Sports and travel
- Unit price: Price of each product in \$
- Quantity: Number of products purchased by customer
- Tax: 5% tax fee for customer buying
- Total: Total price including tax
- Date: Date of purchase (Record available from January 2019 to March 2019)
- Time: Purchase time (10am to 9pm)
- Payment: Payment used by customer for purchase (3 methods are available – Cash, Credit card and Ewallet)
- Rating: Customer stratification rating on their overall shopping experience (On a scale of 1 to 10)

Steps Taken in Our Dataset

- Exploration
 - a. See if there missing values and ensure data type

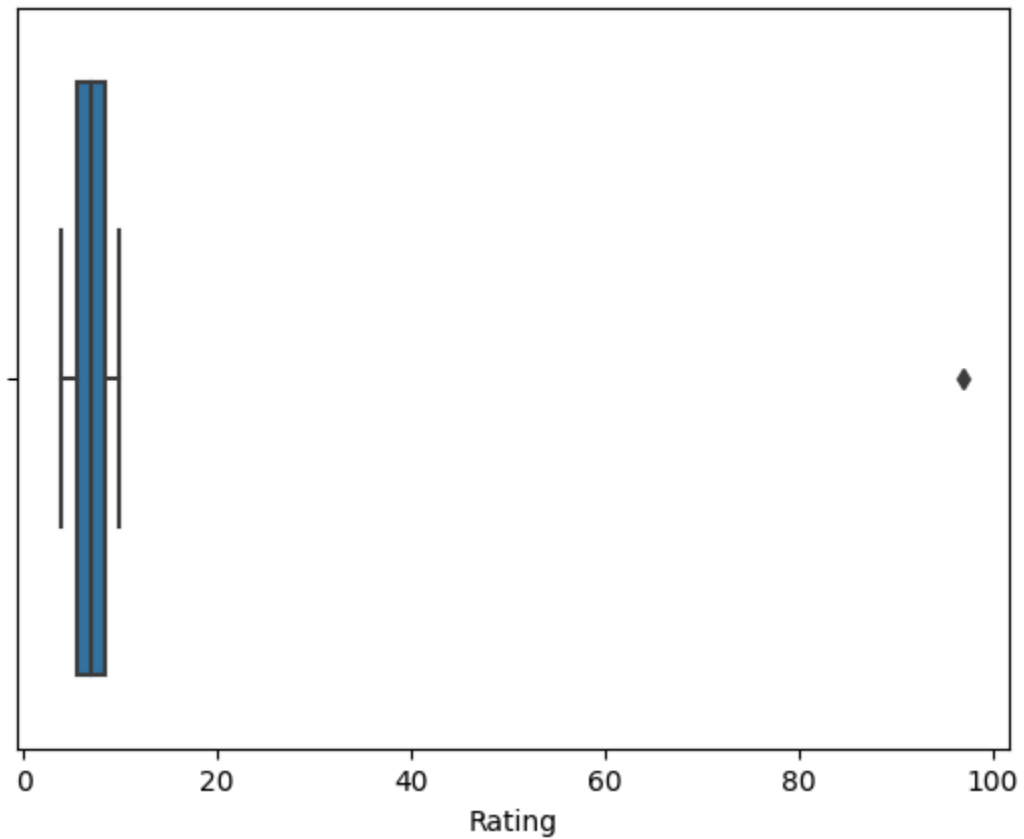
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1006 entries, 0 to 1005
Data columns (total 16 columns):
#   Column              Non-Null Count  Dtype
---  -
0   Invoice ID           1006 non-null   object
1   Branch              1006 non-null   object
2   Yangon              1006 non-null   int64
3   Naypyitaw           1006 non-null   int64
4   Mandalay            1006 non-null   int64
5   Customer type       1006 non-null   object
6   Gender              1006 non-null   object
7   Product line        1006 non-null   object
8   Unit price          1006 non-null   object
9   Quantity            1006 non-null   int64
10  Tax 5%              997 non-null    float64
11  Total               1003 non-null   float64
12  Date                1006 non-null   object
13  Time                1006 non-null   object
14  Payment             1006 non-null   object
15  Rating              1006 non-null   float64
dtypes: float64(3), int64(4), object(9)
```

- b. Explore the columns that make 'Unit price' object

Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time
Normal	Female	Electronic accessories	12.45 USD	6	NaN	78.4350	2/9/2019	13:11
Normal	Female	Fashion accessories	12.09 USD	-1	NaN	12.6945	1/26/2019	18:19
Normal	Male	Electronic accessories	10.56 USD	-8	NaN	88.7040	1/24/2019	17:43
Member	Female	Fashion accessories	10.18 USD	-8	NaN	85.5120	3/30/2019	12:51
Normal	Male	Food and beverages	11.53 USD	-7	NaN	84.7455	1/28/2019	17:35

The columns have 'USD' in it

c. Check Outliers in Data using boxplot



d. Check data in time

Customer type	Gender	Product line	Unit price	Quantity	Tax 5%	Total	Date	Time
Normal	Male	Health and beauty	58.22	8	NaN	489.048	1/27/2019	8 - 30 PM

e. See duplicated data

```
num_duplicates = df.duplicated().sum()

print(f"Number of duplicated rows: {num_duplicates}")
```

Number of duplicated rows: 6

f. See 'Quantity' Column

Customer type	Gender	Product line	Unit price	Quantity
Normal	Female	Fashion accessories	12.09 USD	-1
Normal	Male	Electronic accessories	10.56 USD	-8
Member	Female	Fashion accessories	10.18 USD	-8
Normal	Male	Food and beverages	11.53 USD	-7

g. Branch and cities columns

Branch	Yangon	Naypyitaw	Mandalay
A	1	0	0
A	1	0	0
C	0	1	0
A	1	0	0

Issues Found

The following issues were identified in the dataset:

1. **Duplicated Rows:** The dataset contains 6 duplicated rows.
2. **Missing Values:** Missing values are present in the 'Tax' and 'Total' columns.
3. **Irregular Missing Value Representation:** The 'Customer Type' column contains missing values represented as '-!'.
The 'Total' column contains missing values represented as '0'.
4. **Inconsistent Time Format:** The 'Time' column contains a value with the 'pm' word.
5. **Non-standard Time Format:** The 'Time' column includes a value formatted as '8-30' instead of the standard time format.
6. **Incorrect Unit Price Formatting:** Some values in the 'Unit Price' column include the 'USD' word.
7. **Negative Quantity Values:** The 'Quantity' column contains negative values.
8. **Typographical Errors:** Some entries in the 'Customer Type' column are represented as 'Memberr' instead of 'Member'.
9. **Excessive Decimals in Tax:** The 'Tax 5%' column contains values with more than 2 decimal places.
10. **Excessive Decimals in Total:** The 'Total' column contains values with more than 2 decimal places.
11. **Outliers in Quantity:** The 'Quantity' column contains some outliers.

- 12. **Outliers in Tax 5%:** The 'Tax 5%' column contains some outliers.
- 13. **Outliers in Total:** The 'Total' column contains some outliers.
- 14. **Outliers in Rating:** The 'Rating' column contains some outliers.
- 15. **Non-standard Date Format:** Dates are not in a standard format.
- 16. **Column misspelling:** Correct 'Naypyitaw' to the correct name.
- 17. **Multiple columns for one variable:** 'Yangon', 'Naypyidaw' and 'Mandalay'
- 18. **Columns name headers:** Capitalization in columns name

Dealing with Issues

To clean and preprocess the issues dataset, the following steps were taken:

1. Remove Duplicated Rows:

- All duplicated rows were identified and removed using the `\drop_duplicates()\` method.`

```
Delete duplicated rows

Data_clean = Data_clean.drop_duplicates()
```

2. Convert Negative Values to Positive in 'Quantity' Column:

- Negative values in the 'Quantity' column were converted to positive using the `\.abs()\` method.`

Convert negative values to positive ones in "Quantity" column

```
Data_clean['Quantity'] = Data_clean['Quantity'].abs()
```

3. Clean 'Unit Price' Column:

- Removed the 'USD' text from values in the 'Unit Price' column and converted the data type to numeric.

Remove 'USD' in 'Unit price' column and convert the data type to numeric

```
contains_usd = Data_clean['Unit price'].astype(str).str.contains('USD', na=False)  
contains_usd.sum()
```

0

4. Standardize 'Customer Type' Values:

- Replaced misspelled values ('Memberr') with the correct value ('Member') in the 'Customer Type' column.

Replace 'Memberr' with 'Member' in 'Customer type' column

```
Data_clean['Customer type'] = Data_clean['Customer type'].astype(str).str.replace('Memberr', 'Member', regex=False)
```

5. Handle Missing Values:

- Missing values represented by '-' in the 'Customer Type' column were replaced with Nulls, and then rows with missing values were dropped.

- Missing values in the 'Tax 5%' and 'Total' columns were replaced using calculated values based on other columns.

Replace '-' in 'Customer type' column with Nulls

```
Data_clean['Customer type'] = Data_clean['Customer type'].replace('-', np.nan)
```

```
# Drop rows with missing values in the 'Customer type' column
Data_clean = Data_clean.dropna(subset=['Customer type'])
```

```
# Replace missing values in 'Tax' column using the given equation
Data_clean['Tax 5%'] = Data_clean['Tax 5%'].fillna(Data_clean['Unit price'] * Data_clean['Quantity'] * 0.05)
```

```
# Verify replacement
print(f"Missing values in 'Tax 5%': {Data_clean['Tax 5%'].isnull().sum()}")
```

```
Missing values in 'Tax 5%': 0
```

```
# Replace missing values in 'Total' column using the given equation
Data_clean['Total'] = Data_clean['Total'].fillna(Data_clean['Unit price'] * Data_clean['Quantity'] + Data_clean['Tax 5%'])
```

```
# Verify replacement
print(f"Missing values in 'Total': {Data_clean['Total'].isnull().sum()}")
```

```
Missing values in 'Total': 0
```

6. Convert Time to 24-Hour Format:

- Standardized the 'Time' column by converting 12-hour time with 'PM' to a 24-hour format using a custom function.

Remove 'PM' in 'Time' column and Change it to 24 format

```
#function to change to 24 hr format
def convert_time_column(time_str):
    return datetime.strptime(time_str.replace(' - ', ':'), "%I:%M %p").strftime("%H:%M")

#aply the function
pm_rows = Data_clean['Time'].str.contains('PM')
Data_clean.loc[pm_rows, 'Time'] = Data_clean.loc[pm_rows, 'Time'].apply(convert_time_column)
```

7. Remove Outliers:

- Outliers in the 'Total,' and 'Rating' columns were identified using the IQR method. Necessary outliers were removed to maintain data consistency.

The outliers in the Tax 5% seem to be reasonable and don't need to be removed based on the unit price and quantity

```
Q1 = Data_clean['Total'].quantile(0.25)
Q3 = Data_clean['Total'].quantile(0.75)
IQR = Q3 - Q1

# Define outliers based on 1.5 * IQR
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Filter the outliers
outliers = Data_clean[(Data_clean['Total'] < lower_bound) | (Data_clean['Total'] > upper_bound)]

# Display the outlier values
print("Outlier Values based on IQR method:\n", outliers['Total'],outliers['Quantity'],outliers['Product line'])
```

```

Q1 = Data_clean['Rating'].quantile(0.25)
Q3 = Data_clean['Rating'].quantile(0.75)
IQR = Q3 - Q1

# Define outliers based on 1.5 * IQR
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Filter the outliers
outliers = Data_clean[(Data_clean['Rating'] < lower_bound) | (Data_clean['Rating'] > upper_bound)]

# Display the outlier values
print("Outlier Values based on IQR method:\n", outliers['Rating'])

```

```

Outlier Values based on IQR method:
157    97.0
Name: Rating, dtype: float64

```

8. Adjust Decimal Places:

- Rounded values in the 'Tax 5%' and 'Total' columns to two decimal places to standardize formatting.

Make 'Total' and 'Tax %5' have 2 decimal places like the other numeric columns

```

Data_clean['Total']=Data_clean['Total'].round(2)
Data_clean['Tax 5%']=Data_clean['Tax 5%'].round(2)

```

9. Standardize Date Format:

- Converted 'Date' to the standard ISO 8601 format using `pd.to_datetime()`.

Change Date to the Standard Format according to ISO 8601

```
Data_clean['Date'] = pd.to_datetime(Data_clean['Date'], format='%m/%d/%Y', errors='coerce')
```

10. Tidiness Adjustments:

- Combined 'Yangon', 'Naypyitaw', and 'Mandalay' columns into a single 'City' column representing city names.
- Removed the 'Branch' column since it was redundant, with branches being uniquely identified by the 'City'.

Tidiness Issues :

1. 'Yangon', 'Naypyitaw' and 'Mandalay' columns could be represented in one column using the name of the country (Multiple Columns for one value)
2. Remove Branch Column because branch and City have the same meaning 'Yangon' => 'A', 'Naypyitaw' => 'C', and 'Mandalay' => 'B'

```
def city(row):  
    if row['Yangon'] == 1 :  
        return 'Yangon'  
    elif row['Naypyitaw'] == 1:  
        return 'Naypyitaw'  
    else:  
        return 'Mandalay'
```

```
# Remove 'Yangon' , 'Naypyitaw' and 'Mandalay' columns  
Data_clean.drop(columns=['Yangon','Naypyitaw','Mandalay'],inplace=True)
```

Tools Used

The following tools and libraries were used for data cleaning and preprocessing:

1. **Python**: The main programming language used for data manipulation and analysis.

2. **Pandas**:

- Used extensively for data manipulation, cleaning, and analysis tasks.
- Provided functionalities like handling missing data, removing duplicates, filtering outliers, and transforming data.


3. **Matplotlib / Seaborn**

- Used for visualizing data and identifying Outliers during data exploration.

Conclusion

The data cleaning and preprocessing of the `Supermarket Sales` dataset were essential steps to ensure the quality, consistency, and reliability of the data for further analysis. By addressing various issues, including missing values, inconsistent formatting, typographical errors, and outliers, the dataset was transformed into a clean and well-structured form.

This process not only improved the dataset's overall integrity but also enhanced its usability for analytical purposes, such as customer behavior analysis, sales trend identification, and performance evaluation. The cleaned



dataset provides accurate insights that can drive decision-making processes.

Using Python and powerful data manipulation libraries like Pandas, the challenges were effectively managed, demonstrating the importance of thorough data preparation in any analytical workflow. This project highlights the critical role of data cleaning in ensuring the validity of subsequent analyses and the value it adds in drawing meaningful and actionable insights.