

#DigitalSkillFair39

Exploratory Data Analysis

Rafif Bagus Putra Pratama



Import Library & Load Data



```
import pandas as pd

df = pd.read_csv('/content/train.csv')
```

[8] df

	ID	date	meals_served	kitchen_staff	temperature_C	humidity_percent	day_of_week	special_event	past_waste_kg	staff
0	0	2022-12-19	196	13	27.887273	45.362854	0	0	7.740587	
1	1	2023-11-21	244	15	10.317872	64.430475	1	0	42.311779	
2	4	2022-02-01	148	16	27.714300	69.046113	1	0	41.184305	
3	5	2023-03-19	157	19	19.173902	46.292823	6	0	41.543492	
4	6	2022-07-18	297	10	26.375233	79.741064	0	0	26.525097	
...
906	1044	2022-03-29	395	18	17.354199	45.138435	1	0	40.550668	
907	1045	2022-11-27	483	11	24.912137	59.485091	6	0	36.470276	
908	1046	2023-04-12	243	11	28.870945	70.508404	2	0	19.767203	

✓ 0s completed at 8:00 PM

- imports the pandas library as pd, used for data manipulation and analysis.
- Reads the CSV file train.csv from the /content/ directory.
- Converts it into a DataFrame and stores it in the variable df.

Displays the data in a tabular format with columns like:

- ID: Row identifier
- date: Date of the record
- meals_served: Number of meals served
- kitchen_staff: Number of kitchen staff
- temperature_C: Temperature (°C)
- humidity_percent: Humidity percentage
- day_of_week: Day of the week (0 = Monday, 6 = Sunday)





Checking Missing Value

Explanation:

- This command provides a concise summary of the DataFrame df.
- It displays information such as:
 - Total number of rows and columns.
 - Column names and their data types.
 - The number of non-null (filled) values in each column.
 - Memory usage of the DataFrame.

Purpose:

To quickly understand the structure, data types, and completeness of the dataset.

A screenshot of a Jupyter Notebook interface. At the top, a toolbar shows a green checkmark, a play button, and the text 'df.info' with a '0s' timer. Below the toolbar, a document icon is visible. The main area contains the following text:

```
pandas.core.frame.DataFrame.info  
def info(verbose: bool | None=None, buf: WriteBuffer[str] | None=None, max_cols: int | None=None,  
memory_usage: bool | str | None=None, show_counts: bool | None=None) -> None
```

</usr/local/lib/python3.11/dist-packages/pandas/core/frame.py>
Print a concise summary of a DataFrame.

This method prints information about a DataFrame including the index dtype and columns, non-null values and memory usage.

Checking Missing Value

Only the staff_experience column has missing values, with a total of 164 rows.

All other columns (ID, date, meals_served, kitchen_staff, temperature_C, humidity_percent, day_of_week, special_event, past_waste_kg, waste_category, and food_waste_kg) have no missing values (NaN).



```
df.isna().sum()
```

	0
ID	0
date	0
meals_served	0
kitchen_staff	0
temperature_C	0
humidity_percent	0
day_of_week	0
special_event	0
past_waste_kg	0
staff_experience	164
waste_category	0
food_waste_kg	0

dtype: int64

Resolving Missing Value

The code handles missing values in all columns of the DataFrame:

- Object (categorical) columns are filled with the mode (most frequent value).
- Numeric columns are filled with the mean (average value).

Changes are applied directly to the DataFrame.

```
✓ 0s # Mengatasi missing value
      for column in df.columns:
          if df[column].dtype == 'object':
              # Jika kolom bertipe object, isi dengan mode
              df[column].fillna(df[column].mode()[0], inplace=True)
          else:
              # Jika kolom bertipe numerik, isi dengan mean
              df[column].fillna(df[column].mean(), inplace=True)
```

Checking and Resolving Duplicate Data

The code performs the following steps:

Check for Duplicates:

- It uses `df.duplicated().sum()` to identify any duplicate rows in the DataFrame.

Handle Duplicates:

- It applies `df.drop_duplicates()` to remove any duplicates if they exist.

Recheck for Duplicates:

- It verifies again with `df.duplicated().sum()` to ensure no duplicates are left.

Conclusion:

The dataset is already clean and free of duplicates, and the handling process did not alter the data.

```
✓ 0s [16] # Mengecek apakah ada duplicate di seluruh kolom
      check_duplicate = df.duplicated().sum()

      print(f"Jumlah data yang duplikat = {check_duplicate}")

      ➞ Jumlah data yang duplikat = 0

✓ 0s [17] # Handling duplicate
      df = df.drop_duplicates()

✓ 0s [18] # Mengecek duplicate setelah di-handle
      handle_duplicate = df.duplicated().sum()

      print(f"Jumlah data yang duplikat = {handle_duplicate}")

      ➞ Jumlah data yang duplikat = 0
```



Terima Kasih Atas

Rafif Bagus Putra Pratama

