DATA CLEANING

Using

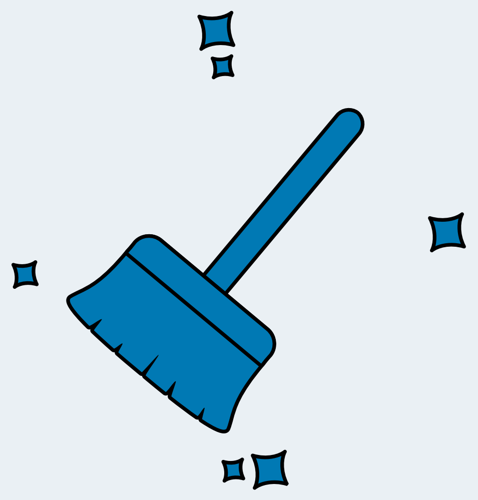
Presented By

ELACROUCH Imane

Plan

- ✓ Definition
- ✓ Why Data Cleaning is Essential
- ✓ Processus of Data Cleaning
- ✓ Hands-On Implementation

Definition



Data cleaning, also known as data cleansing or data scrubbing, is the process of identifying and correcting errors or inconsistencies in datasets to improve their quality.

Why Data Cleaning is Essential

Cleaning data is crucial because it ensures that the information you have is accurate and reliable.

When you clean data, you fix errors and inconsistencies, making the data more trustworthy.

Why Data Cleaning is Essential

This is important for making informed decisions, conducting accurate analyses, and avoiding misunderstandings or mistakes that could arise from using flawed data.

Plan

- ✓ Definition
- ✓ Why Data Cleaning is Essential
- ✓ Processus of Data Cleaning
- ✓ Hands-On Implementation

Processus

IMPORTING DATA

DATA MEASURES

MISSING VALUES

DUPLICATES DATA

IMPORTING DATA

In Python, various libraries are available for importing and handling data, with "**pandas**" being one of the most commonly used.

Pandas offers functions to read data from diverse file formats.

Excel files

JSON files

SQL databases

....

import a **CSV** file using pandas

```
df = pd.read_csv('your_file.csv')
```

DATA MEASURES

info()

Provides a concise summary of the DataFrame, including the data types, non-null values, and memory usage. Useful for quickly assessing the structure of the dataset.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
```

DATA MEASURES

describe()

Generates descriptive statistics of the numeric columns in the DataFrame, including count, mean, standard deviation, minimum.....

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

DATA MEASURES

dtypes

Returns the data types of each column in the DataFrame. Useful for understanding the data type of each variable.

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

DATA MEASURES

shape

Returns a tuple representing the dimensions of the DataFrame (number of rows, number of columns). Useful for quickly checking the size of your dataset.

```
df.shape
```

```
(891, 12)
```

DATA MEASURES

head()

Displays the first n rows of the DataFrame. By default, it shows the first 5 rows. Useful for getting a quick overview of the dataset.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

DATA MEASURES

tail()

Displays the last n rows of the DataFrame. By default, it shows the last 5 rows. Useful for inspecting the end of the dataset.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.00	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.00	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.45	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.00	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	NaN	Q

MISSING VALUES

Detecting

```
missing_values = df.isnull().sum()
print(missing_values)
```

- The isnull() method returns a DataFrame of the same shape as the input with True and False indicating the presence of missing values

- The sum() method can be used to count the missing values for each column

MISSING VALUES

Removing

Columns with Missing Values

Rows with Missing Values

```
df_cleaned = df.drop('column_name', axis=1)
```

```
df_cleaned = df.dropna()
```

MISSING VALUES

Replacing

You can fill missing values in an entire dataset using the mean value of each column.

```
mean_values = df.mean()
df.fillna(mean_values, inplace=True)
```

MISSING VALUES

For time series data, you may want to use interpolation methods to estimate missing values based on the existing data.

```
df['column_name'].interpolate(method='linear', inplace=True)
```

DUPLICATES DATA

Duplicate data refers to identical or very similar records within a dataset. Identifying and handling duplicate data is crucial for maintaining data accuracy and avoiding biases in analyses.

Detecting

```
duplicates = df[df.duplicated()]
```

DUPLICATES DATA

Removing

```
df.drop_duplicates(inplace=True)
```

REMARK

In datasets, it is common also to encounter incorrect data, outliers, and other anomalies.

Incorrect Data

dates
24/06/2024
15/04/2020
14 Feb 2021

Incorrect data examples include instances where there is a wrong syntax for dates.

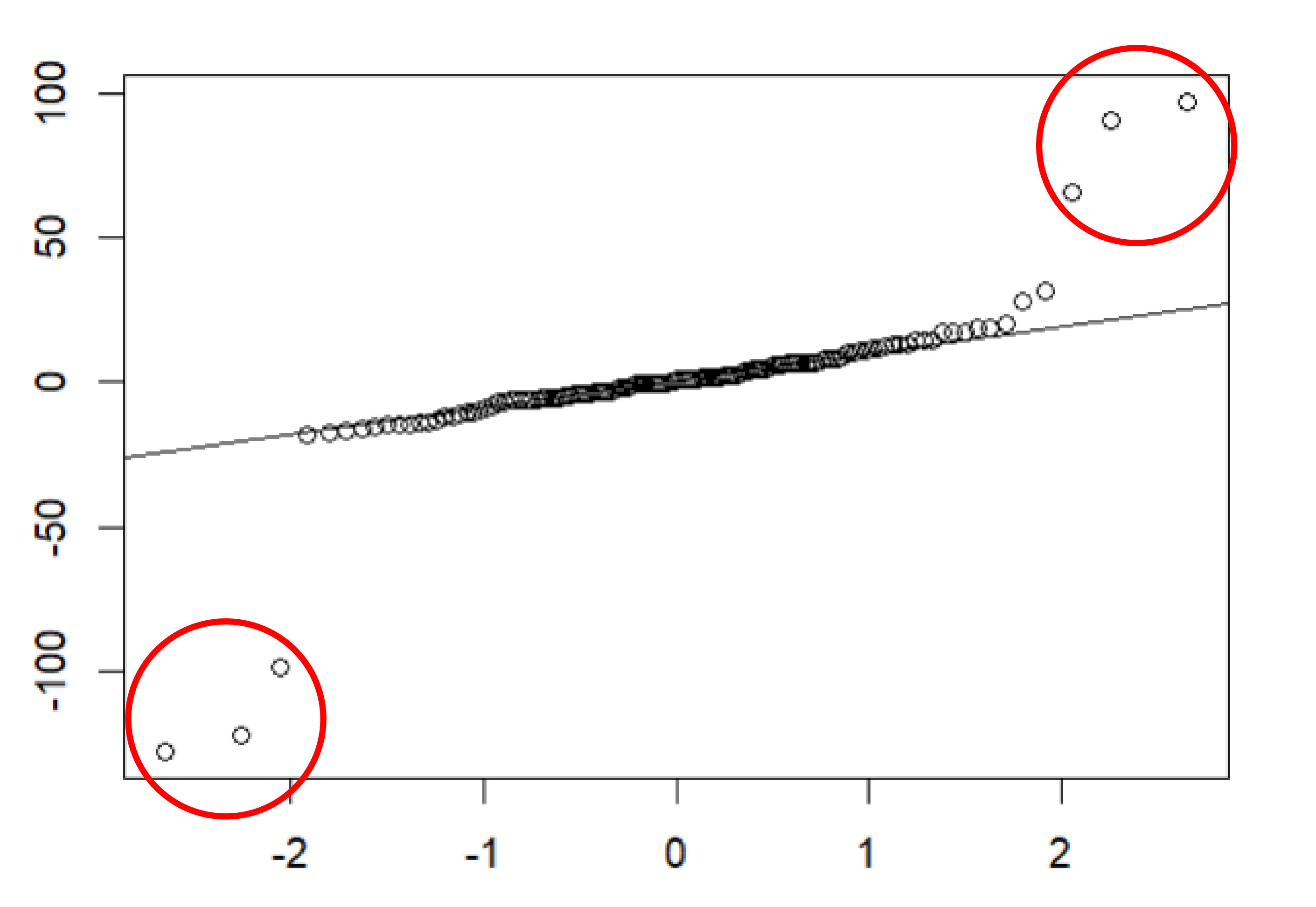
This how we can fix

```
# Replacing the specific date in the DataFrame
df.at[2, 'dates'] = pd.to_datetime(df.at[2, 'dates'], errors='coerce', format='%d %b %Y').strftime('%d/%m/%Y')
```

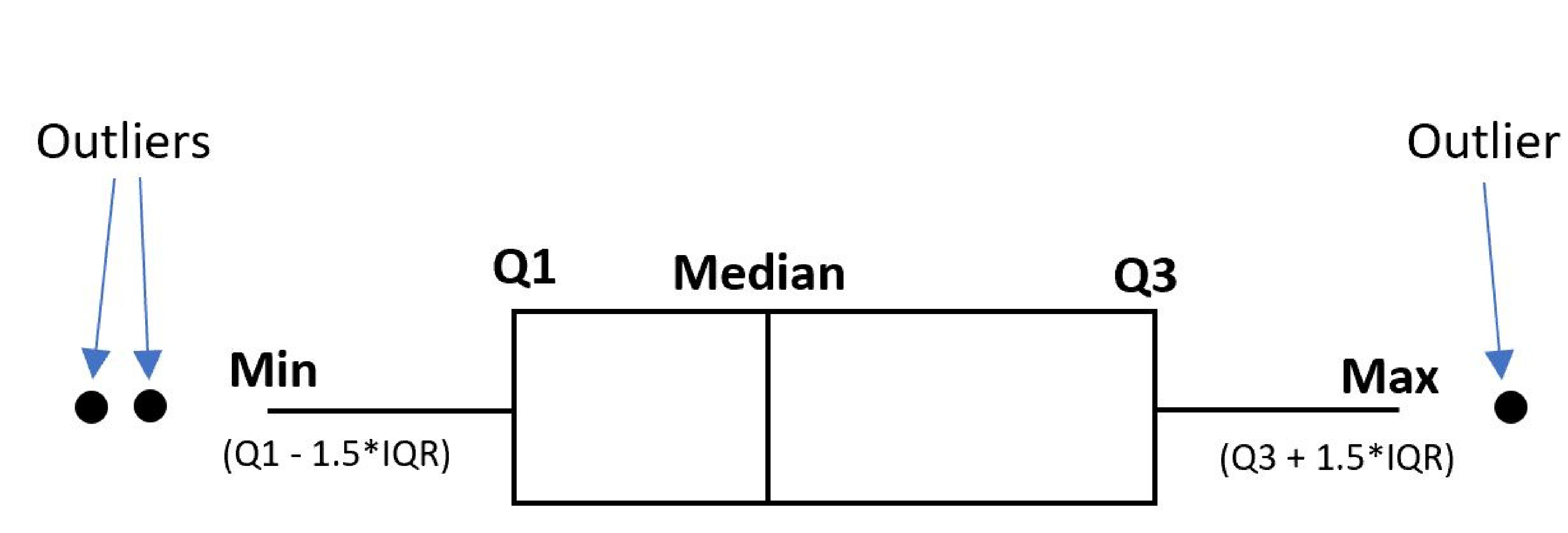
dates
24/06/2024
15/04/2020
14/02/2021

Outliers

Outliers are data points that significantly deviate from the majority of the dataset, often to the extent that they can impact the overall analysis or statistical measures.



Box Plot



HOW TO REMOVE Outliers

```python
# Function to remove outliers using IQR
def remove_outliers(df, column_name):
    Q1 = df[column_name].quantile(0.25)
    Q3 = df[column_name].quantile(0.75)
    IQR = Q3 - Q1

    # Defining the lower and upper bounds for outliers
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Removing outliers
    df_filtered = df[(df[column_name] >= lower_bound) & (df[column_name] <= upper_bound)]

    return df_filtered
```

REMARK

There are instances where the removal of outliers may not be crucial or essential because they might contain meaningful information or represent unique circumstances within the data.

Plan

- ✓ Definition
- ✓ Why Data Cleaning is Essential
- ✓ Processus of Data Cleaning
- ✓ Hands-On Implementation

In this hands-on implementation, we will leverage the previously outlined data cleaning processes, thus we delve into data analysis and predictive modeling using the infamous Titanic dataset.

THANK YOU FOR YOUR ATTENTION