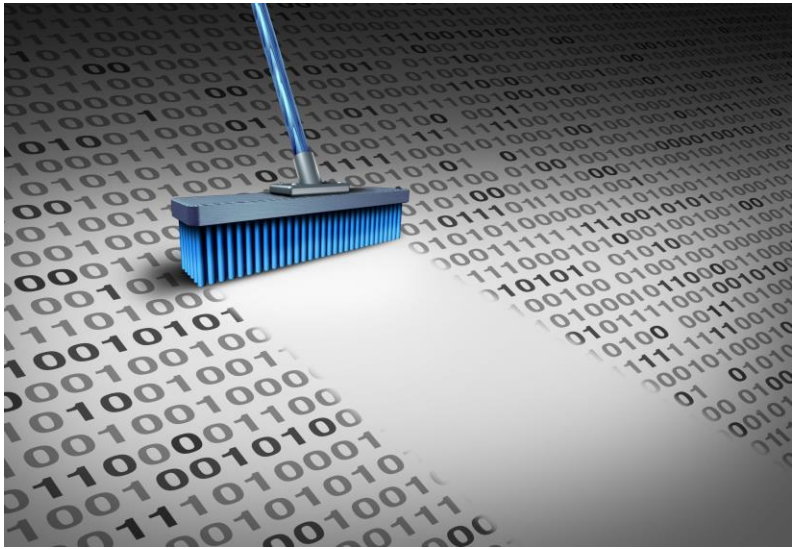
Workshop on Python (Day 6)

By Suriya G
Organized by Suresh Sir, UPNM

TABLE OF CONTENTS

What is Data Cleaning?

- Act of detecting & removing inaccurate records from the data.
- Sometimes replacing, modifying or even deleting them plays an important role in analysis



Why is Data Dirty?

- Dummy Values
- Absence of Data
- Violating business rules
- Multipurpose fields
- Data Repetition
- Typo Errors

Tools for cleaning

Basic:

- Strings, lists, loops.

Pandas:

- Fast DataFrame operations.

Handling Missing Values

- **Category comparison**
- Used for comparing different categories or groups
- Example: Sales analysis across regions

missing.py

```
1  # Basic syntax for bar chart
2  df.fillna(value)          #Filling gaps
3  Df.dropna()               #remove rows
```

Removing Duplicates

- **Category comparison**
- Used for comparing different categories or groups
- Example: Sales analysis across regions

missing.py

```
1  # Basic syntax for bar chart
2  df.duplicated()           #Find repeats
3  df.drop_duplicates()      #remove rows
```

Cleaning Strings

- **Category comparison**
- Used for comparing different categories or groups
- Example: Sales analysis across regions

missing.py

```
1  # Basic syntax for bar chart
2  df.str.lower()          #Fix case
3  df.str.strip()          #Trim Spaces
```

Converting Data Types

- **Category comparison**
- Used for comparing different categories or groups
- Example: Sales analysis across regions

missing.py

```
1  # Basic syntax for bar chart
2  df.astype(int)              #Convert object to integer
```

Converting Data Types

- **Category comparison**
- Used for comparing different categories or groups
- Example: Sales analysis across regions

missing.py

```
1  # Basic syntax for bar chart
2  df.astype(int)              #Convert object to integer
```

Lets Practice with new dataset

Handling Missing Values

missing.py

```
1  import pandas as pd
2
3  df = pd.read_csv("customers.csv")
4
5  # Fill missing ages with mean
6  df["age"] = df["age"].fillna(df["age"].mean())
7
8  # Check result
9  print("After filling missing ages:")
10 print(df[["name", "age"]])
```

Removing Duplicates

duplicates.py

```
1  # Check duplicates
2  print("Duplicates:")
3  print(df[df.duplicated()])
4
5  # Remove duplicates
6  df = df.drop_duplicates()
7
8  # Check result
9  print("\nAfter removing duplicates:")
10 print(df)
```

Cleaning Strings

duplicates.py

```
1  # Check duplicates
2  print("Duplicates:")
3  print(df[df.duplicated()])
4
5  # Remove duplicates
6  df = df.drop_duplicates()
7
8  # Check result
9  print("\nAfter removing duplicates:")
10 print(df)
```

Converting Data Types

dtypes.py

```
df["purchase"] = pd.to_numeric(df["purchase"],
errors="coerce") # Check result
print("After converting purchases:")
print(df[["name", "purchase"]])
```

Converting Data Types

dtypes.py

```
df["purchase"] = pd.to_numeric(df["purchase"],
errors="coerce") # Check result
print("After converting purchases:")
print(df[["name", "purchase"]])
```

Filtering Bad Data

dtypes.py

```
df = df.loc[(df["age"] >= 0) |
(df["age"].isnull())] df =
df.dropna(subset=["purchase"])
print("After filtering bad data:")
print(df)
```

Validating Age

dtypes.py

```
1  if (df["age"] < 0).any():
2      print("Warning: Negative ages found!")
3  else:
4      print("Age validation passed.")
5
6
```