

Disclaimer

- This presentation is purely for academic purpose and does not carry any commercial value.
- All images and photos used in this presentation are property of respective image holder(s) and due credit is provided to them. Images are used only for indicative purpose and does not carry any other meaning.
- All information and data in this slide are collected from open domain.

MANISH GODSE, Ph.D.(IIT Bombay)

Welcome

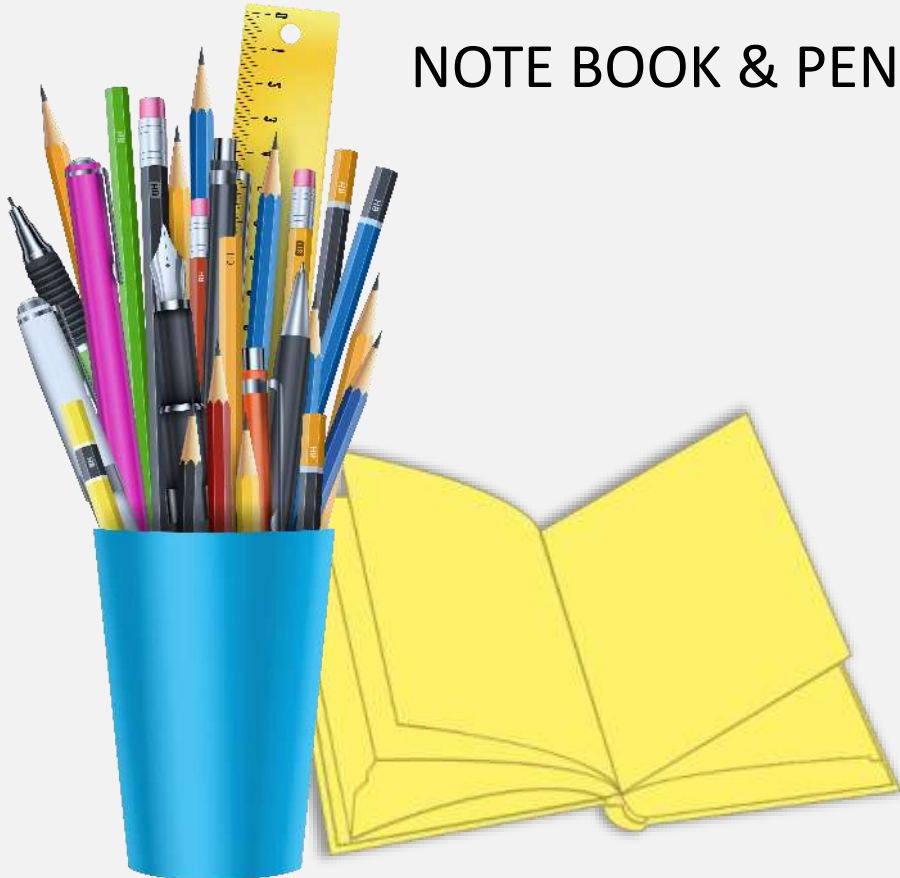


Request & Instructions



PLEASE OPEN

NOTE BOOK & PEN



CALCULATOR



LAPTOP OR DESKTOP,
IF YOU HAVE.



PLEASE FOLLOW THIS

SILENCE



MUTE
MIC



RAISE HAND



NO
CHAT



SILENT
MODE



DATA CLEANING



BOOKS & REFERENCES

- https://scikit-learn.org/stable/data_transforms.html
- <https://machinelearningmastery.com/model-based-outlier-detection-and-removal-in-python/>

Table of Contents

1. Loading Dataset
2. Check Data
3. Data Cleaning Tasks
4. Manage Duplicate Records
5. Manage Missing Data
6. Remove Missing Data
7. Replace or Impute Missing Data
8. Manage Outliers



LOADING DATASET

1



LOADING A DATASET

FILE FORMAT

Data file may be in **EXCEL** or **CSV** format.

COLUMN NAMES OR HEADERS

It may or not have column **names** or **headers**. These names are variable names.

PYTHON STATEMENT TO LOAD FILE

The below Python statement is used to read the file in CSV and EXCEL format.

```
df = pd.read_csv("file_path/data_file.csv")
```

```
df = pd.read_excel("file_path/data_file.xlsx")
```

HEADER IN DATASET

- The **column names** or **headers** are inferred from the **first line** of the **file**. This is done by using the keyword "**header**".
- The default value is **header=0**. Hence it is not necessary to mention it, if the file has headers in **first row**.
- The Python statement **with header** is as below.

```
df = pd.read_csv("file_path/data_file.csv" ", header=0)  
df = pd.read_excel("file_path/data_file.xlsx" ", header=0)
```
- The Python statement **without header** is as below.

```
df = pd.read_csv("file_path/data_file.csv" ", header=None)  
df = pd.read_excel("file_path/data_file.xlsx" ", header=None)
```

EXAMPLE – header=0

import pandas library

```
import pandas as pd
```

Get file path

```
sales_dataset = "E:/Courses/Machine_Learning_Python/data/Super_Store_Sales.csv"
```

Read file and save as DataFrame

```
df_sales = pd.read_csv(sales_dataset, header=0)
```

Check column heads or names

```
print(df_sales.columns)
```

Get count of rows and column

```
print(df_sales.shape)
```


EXAMPLE – header=None

import pandas library

```
import pandas as pd
```

Get file path

```
sales_dataset = "E:/Courses/Machine_Learning_Python/data/Super_Store_Sales.csv"
```

Read file and save as DataFrame

```
df_sales = pd.read_csv(sales_dataset, header=None)
```

Check column heads

```
print(df_sales.columns)
```

Get count of rows and column

```
print(df_sales.shape)
```

CHECK DATA

2



CHECK DATA (1/2)

import pandas library

```
import pandas as pd
```

Get file path

```
data = "E:/Courses/Machine_Learning_Python/data/MonthlySalesNone.csv"
```

Read file and save as DataFrame

```
df = pd.read_csv(data, header=0)
```

Print the dataset

```
print(df)
```

Get count of rows and column

```
print(df.shape)
```

CHECK DATA (2/2)

Check column heads

```
print(df.columns)
```

Print datatype

```
print(df.dtypes)
```

View data

```
print(df)
```

View top 5 data

```
print(df.head(5))
```

View bottom 5 data

```
print(df.tail(5))
```


DATA CLEANING TASKS

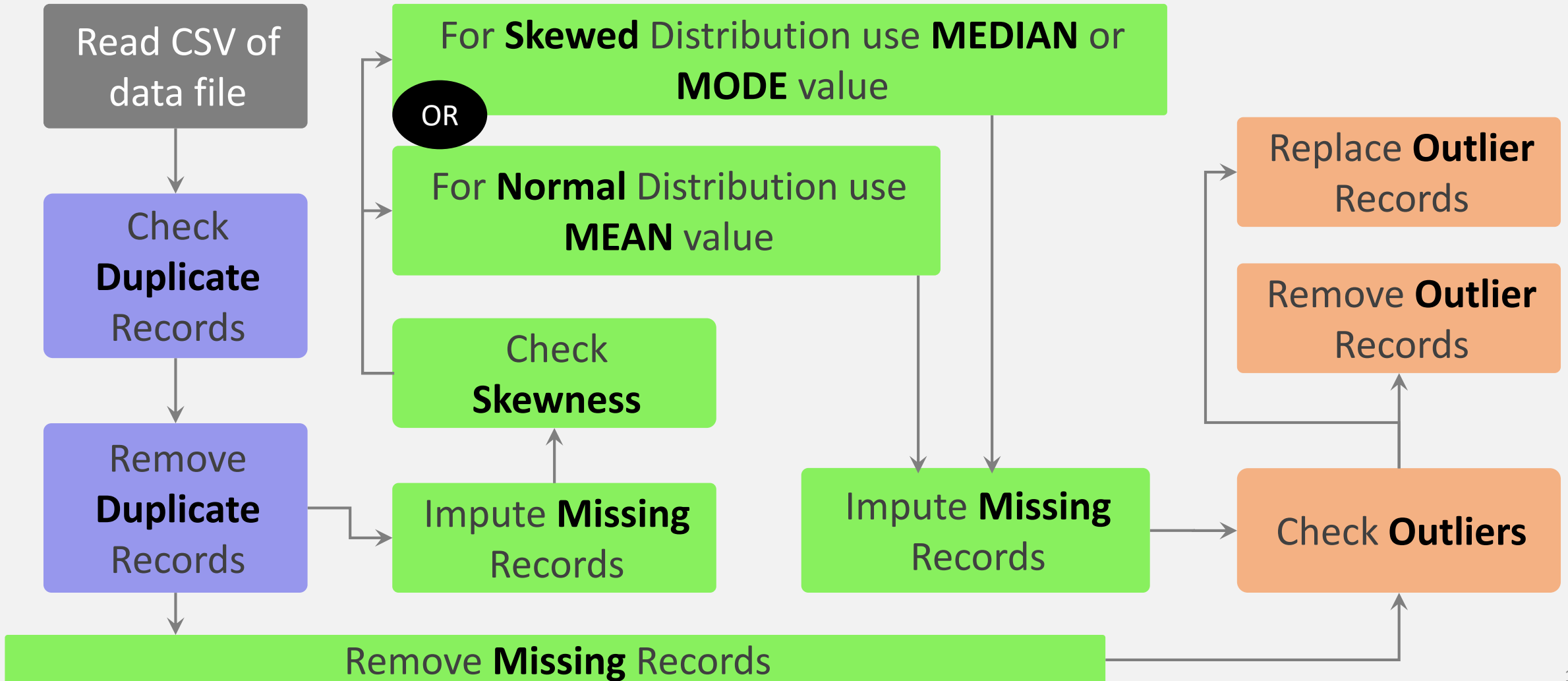
3



DATA CLEANING TASKS

Data cleaning tasks are as below.

1. **Duplicate**
2. **Missing Value**
3. **Outlier**



CHECK DATA (1/2)

import pandas library

```
import pandas as pd
```

Get file path

```
data = "E:/Courses/Machine_Learning_Python/data/MonthlySalesNone.csv"
```

Read file and save as DataFrame

```
df = pd.read_csv(data, header=0)
```

Print the dataset

```
print(df)
```

Get count of rows and column

```
print(df.shape)
```

CHECK DATA (2/2)

Check column heads

```
print(df.columns)
```

Print datatype

```
print(df.dtypes)
```

View data

```
print(df)
```

View top 5 data

```
print(df.head(5))
```

View bottom 5 data

```
print(df.tail(5))
```


MANAGE DUPLICATE RECORDS

4



READ FILE

import pandas library

```
import pandas as pd
```

Get file path

```
sales_data =  
"E:/Courses/Machine_Learning_Python/data/MonthlySales_DuplicateData.csv"
```

Read file and save as DataFrame

```
df = pd.read_csv(sales_data, header=0)  
print('Original DataFrame of Input File\n',df)
```

CHECK DUPLICATE RECORDS

Get all duplicate rows based on all columns

```
print('Duplicate records = ')\nprint(df[df.duplicated()])
```

Get count of duplicate rows

```
print('\nCount of duplicate records = ',df.duplicated().sum())
```

Get row count of data

```
count_row = df.shape[0]\nprint('\nNumber of records including duplicate records = ',count_row)
```

REMOVE DUPLICATE RECORDS

% of duplicate rows

```
rec = 100*df.duplicated().sum()/count_row  
print('\nPercentage of duplicate records = ',rec)
```

Remove duplicate rows

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

Recheck duplicate rows

```
print('\nRecheck count of duplicate records after removal of duplicate records =  
,df.duplicated().sum())
```

MANAGE MISSING DATA

5



MISSING DATA TREATMENT

REMOVE (OR TRIM) Missing Data

- The **dropna()** function is used to remove row having NaN or missing values.
- This method is advisable when very limited data has NaN. Removing missing data rows may lead to loss of information.

IMPUTE (or REPLACE) Missing Data

- Methods such as **mean()**, **median()** and **mode()** can be used on Dataframe for imputing missing values. However, choice of it depend on data distribution.
- Mean value is used to replace the missing value, when the data distribution is symmetric. Median or mode is used for skewed data distribution.
- Pandas Dataframe method **fillna()** can be used along with **mean()**, **median()** and **mode()** to replace the missing values.

REMOVE MISSING DATA

6



READ FILE

import pandas library

```
import pandas as pd
```

Get the file path

```
data = "E:/Courses/Machine_Learning_Python/data/MonthlySales_MissingData.csv"
```

Read the file and save as DataFrame

```
df = pd.read_csv(data, header=0)
```

```
print('Original DataFrame of input file',df)
```

COUNT MISSING (OR NONE) DATA

Get count of missing values

```
print('Count of missing value in each column =')  
print(df.isnull().sum())
```

Calculate count of missing values

```
sum_NULL = df.isnull().sum()
```

Calculate % of missing values

```
perc_NULL = (sum_NULL/df.isnull().count())*100  
missing_data = pd.DataFrame({'Missing Data': sum_NULL, '% Missing Data': perc_NULL})  
print('\n% of missing values = \n')  
print(missing_data)
```

REMOVE MISSING DATA

Remove the rows where at least one element is missing

```
df.dropna()
```

Get the count of missing values in each column

```
sum_NULL = df.isnull().sum()
```

```
print("\nCount of missing values in each column = ", sum_NULL)
```

```
print("\nNumber of rows before removing missing values = ", df.shape[0])
```

```
print("\nNumber of rows after removing missing values = ", df_new.shape[0])
```

```
print("\nNumber of rows removed = ", df.shape[0] - df_new.shape[0])
```


REPLACE OR IMPUTE MISSING DATA

7



READ FILE

import pandas library

```
import pandas as pd
```

Get the file path

```
data = "E:/Courses/Machine_Learning_Python/data/MonthlySales_MissingData.csv"
```

Read the file and save as DataFrame

```
df = pd.read_csv(data, header=0)
```

```
print('Original DataFrame of Input File\n',df)
```

COUNT MISSING DATA

Get count of missing values

```
print('\nCount of missing value in each column =')  
print(df.isnull().sum())
```

Calculate count of missing values

```
sum_NULL = df.isnull().sum()
```

Calculate % of missing values

```
perc_NULL = (sum_NULL/df.isnull().count())*100  
missing_data = pd.DataFrame({'Missing Data': sum_NULL, '% Missing Data': perc_NULL})  
print('\n% of missing values = \n')  
print(missing_data)
```

CALCULATE SKEWNESS & MEDIAN

Calculate skewness of sales by skipping missing values

```
print('\nSkewness = ', df['sales'].skew(skipna = True))
```

Calculate median value

```
med = df['sales'].median()
```

```
print('\nMedian = ', med)
```

REPLACE MISSING DATA

Fill NaN values in sales column by mean value or Replace missing value by mean

```
df['sales'] = df['sales'].fillna(med)
```

Get the count of missing values in each column

```
print('\nCount of missing value in each column =')
```

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


MANAGE OUTLIERS

8



READ FILE

import pandas library

```
import pandas as pd
```

Get file path

```
data = "E:/Courses/Machine_Learning_Python/data/MonthlySales_OutlierData.csv"
```

Read file and save as DataFrame

```
df = pd.read_csv(data, header=0)
```

```
print('Original DataFrame of input file',df)
```

COUNT MISSING DATA

Get count of missing values

```
print('\nCount of missing value in each column =')  
print(df.isnull().sum())
```

Calculate count of missing values

```
sum_NULL = df.isnull().sum()
```

Calculate % of missing values

```
perc_NULL = (sum_NULL/df.isnull().count())*100  
missing_data = pd.DataFrame({'Missing Data': sum_NULL, '% Missing Data': perc_NULL})  
print('\n% of missing values = \n')  
print(missing_data)
```

CALCULATE SKEWNESS & MEDIAN

Calculate skewness of sales by skipping missing values

```
print('\nSkewness = ', df['sales'].skew(skipna = True))
```

Calculate median value

```
med = df['sales'].median()
```

```
print('\nMedian = ', med)
```

REPLACE MISSING DATA

Fill NaN values in sales column by mean value or Replace missing value by mean

```
df['sales'] = df['sales'].fillna(med)
```

Get the count of missing values in each column

```
print('\nCount of missing value in each column =')
```

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

Z-Score can not be calculated, if any of the values is missing or NULL or NaN. If there is any missing value then it has to be removed before calculation of Z-Score.

Z-SCORE FOR OUTLIER DETECTION (1/2)

Import numpy library

```
import numpy as np
```

Import library to calculate z-score

```
from scipy import stats
```

Calculate z-score

```
z_scores = stats.zscore(df['sales'])
```

Get the absolute value of z-score

```
abs_z = np.abs(z_scores)
```

Z-SCORE FOR OUTLIER DETECTION (2/2)

Create dataframe of absolute value of z-score

```
sp = pd.DataFrame(abs_z, columns=['z_score'])  
print('\nValues of z-score = \n', sp)
```

Print dataframe of outliers. Threshold value for outliers is above 3.

```
print('\nOutlier data \n', sp[abs_z > 3])
```

COMBINE Z-SCORE WITH ORIGINAL DATA

Combine z-score data with original dataframe, and create new dataframe

```
df_col = pd.concat([df,sp], axis=1)
print('\nCombined dataframe \n', df_col)
```

Calculate median value

```
med = df['sales'].median()
print('\nMedian = ', med)
```

Replace outlier with median

```
df_col['sales'].mask((df_col['z_score'] > 3), med, inplace=True)
print('\nNew dataframe after outliers are replaced with median. \n', df_col)
```

REMOVE Z-SCORE COLUMN FROM DATAFRAME

Remove column name 'z_score'

```
df_col.drop(['z_score'], axis = 1)
```

QUESTION AND ANSWERS

