

# CS-23334 FUNDAMENTALS OF DATA SCIENCE

**ABENANTHAN P 240701005**

**Experiment No: 3**

**Date: 07.08.2025**

## **3.A Handling Missing Data In a Dataset**

### **Aim:**

To demonstrate an experiment to handle missing data and inappropriate data using pandas

### **Algorithm:**

*Step 1: Identify Missing Data*

*Step 2: Quantify and Visualize Missingness*

*Step 3: Decide on a Strategy (Drop, Impute, Flag)*

*Step 4: Apply the Chosen Method and Validate*

### **Given Dataset:**

CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill	NoOfPax	EstimatedSalary	Age_Group
1	20-25	4	Ibis	veg	1300	2	40000	20-25
2	30-35	5	LemonTree	Non-Veg	2000	3	59000	30-35
3	25-30	6	RedFox	Veg	1322	2	30000	25-30
4	20-25	-1	LemonTree	Veg	1234	2	120000	20-25
5	35+	3	Ibis	Vegetarian	989	2	45000	35+
6	35+	3	Ibys	Non-Veg	1909	2	122220	35+
7	35+	4	RedFox	Vegetarian	1000	-1	21122	35+
8	20-25	7	LemonTree	Veg	2999	-10	345673	20-25
9	25-30	2	Ibis	Non-Veg	3456	3	-99999	25-30
9	25-30	2	Ibis	Non-Veg	3456	3	-99999	25-30
10	30-35	5	RedFox	non-Veg	6755	4	87777	30-35

## About Dataset:

*No.of Columns* =9 (called as series – Customer ID, Age Group, Rating(1-5),Hotel, Food Preference, Bill, No Of Pax, Estimated Salary)

*CutomerID*: Numerical Continuous data

*Age*: Categorical Data

*Rating (1-5)*: Numerical Discrete Data

*Hotel*: Categorical Data

*Food*: Categorical Data

*Bill*: Numerical Continuous data

*NoOfPax*: Numerical Discrete

*EstimatedSalary*: Numerical Continuous data

## Code with Output:

```
import numpy as np
import pandas as pd
df=pd.read_csv(r"D:\REC 2nd Year\Data Science\Data Sets\
Hotel_Dataset.csv")
df
```

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
0	1	20-25	4	Ibis	veg	1300
1	2	30-35	5	LemonTree	Non-Veg	2000
2	3	25-30	6	RedFox	Veg	1322
3	4	20-25	-1	LemonTree	Veg	1234
4	5	35+	3	Ibis	Vegetarian	989
5	6	35+	3	Ibys	Non-Veg	1909
6	7	35+	4	RedFox	Vegetarian	1000
7	8	20-25	7	LemonTree	Veg	2999
8	9	25-30	2	Ibis	Non-Veg	3456
9	9	25-30	2	Ibis	Non-Veg	3456
10	10	30-35	5	RedFox	non-Veg	-6755

```
df.duplicated()
```

```
0    False
1    False
2    False
3    False
4    False
```

```
5    False
6    False
7    False
8    False
9     True
10   False
dtype: bool
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11 entries, 0 to 10
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   CustomerID            11 non-null    int64
1   Age_Group             11 non-null    object
2   Rating(1-5)          11 non-null    int64
3   Hotel                 11 non-null    object
4   FoodPreference        11 non-null    object
5   Bill                 11 non-null    int64
6   NoOfPax              11 non-null    int64
7   EstimatedSalary      11 non-null    int64
8   Age_Group.1          11 non-null    object
dtypes: int64(5), object(4)
memory usage: 924.0+ bytes
```

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

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
0	1	20-25	4	Ibis	veg	1300
1	2	30-35	5	LemonTree	Non-Veg	2000
2	3	25-30	6	RedFox	Veg	1322
3	4	20-25	-1	LemonTree	Veg	1234
4	5	35+	3	Ibis	Vegetarian	989
5	6	35+	3	Ibys	Non-Veg	1909
6	7	35+	4	RedFox	Vegetarian	1000
7	8	20-25	7	LemonTree	Veg	2999
8	9	25-30	2	Ibis	Non-Veg	3456
10	10	30-35	5	RedFox	non-Veg	-6755

```
len(df)

10

index=np.array(list(range(0,len(df))))
df.set_index(index,inplace=True)
index
```

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
df
```

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
0	1	20-25	4	Ibis	veg	1300
2						
1	2	30-35	5	LemonTree	Non-Veg	2000
3						
2	3	25-30	6	RedFox	Veg	1322
2						
3	4	20-25	-1	LemonTree	Veg	1234
2						
4	5	35+	3	Ibis	Vegetarian	989
2						
5	6	35+	3	Ibys	Non-Veg	1909
2						
6	7	35+	4	RedFox	Vegetarian	1000
-1						
7	8	20-25	7	LemonTree	Veg	2999
-10						
8	9	25-30	2	Ibis	Non-Veg	3456
3						
9	10	30-35	5	RedFox	non-Veg	-6755
4						

```
df.drop(['Age_Group.1'],axis=1,inplace=True)
df
```

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
0	1	20-25	4	Ibis	veg	1300
2						
1	2	30-35	5	LemonTree	Non-Veg	2000
3						
2	3	25-30	6	RedFox	Veg	1322
2						
3	4	20-25	-1	LemonTree	Veg	1234
2						
4	5	35+	3	Ibis	Vegetarian	989
2						
5	6	35+	3	Ibys	Non-Veg	1909
2						
6	7	35+	4	RedFox	Vegetarian	1000
-1						
7	8	20-25	7	LemonTree	Veg	2999
-10						
8	9	25-30	2	Ibis	Non-Veg	3456
3						
9	10	30-35	5	RedFox	non-Veg	-6755
4						

```
df.CustomerID.loc[df.CustomerID<0]=np.nan
df.Bill.loc[df.Bill<0]=np.nan
```

```
df.EstimatedSalary.loc[df.EstimatedSalary<0]=np.nan
df
```

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
0	1.0	20-25	4	Ibis	veg	1300.0
1	2.0	30-35	5	LemonTree	Non-Veg	2000.0
2	3.0	25-30	6	RedFox	Veg	1322.0
3	4.0	20-25	-1	LemonTree	Veg	1234.0
4	5.0	35+	3	Ibis	Vegetarian	989.0
5	6.0	35+	3	Ibys	Non-Veg	1909.0
6	7.0	35+	4	RedFox	Vegetarian	1000.0
7	8.0	20-25	7	LemonTree	Veg	2999.0
8	9.0	25-30	2	Ibis	Non-Veg	3456.0
9	10.0	30-35	5	RedFox	non-Veg	NaN

```
df['NoOfPax'].loc[(df['NoOfPax']<1) | (df['NoOfPax']>20)]=np.nan
df
```

	CustomerID	Age_Group	Rating(1-5)	Hotel	FoodPreference	Bill
0	1.0	20-25	4	Ibis	veg	1300.0
1	2.0	30-35	5	LemonTree	Non-Veg	2000.0
2	3.0	25-30	6	RedFox	Veg	1322.0
3	4.0	20-25	-1	LemonTree	Veg	1234.0
4	5.0	35+	3	Ibis	Vegetarian	989.0
5	6.0	35+	3	Ibys	Non-Veg	1909.0
6	7.0	35+	4	RedFox	Vegetarian	1000.0
7	8.0	20-25	7	LemonTree	Veg	2999.0
8	9.0	25-30	2	Ibis	Non-Veg	3456.0
9	10.0	30-35	5	RedFox	non-Veg	NaN

```

df.Age_Group.unique()
array(['20-25', '30-35', '25-30', '35+'], dtype=object)

df.Hotel.unique()
array(['Ibis', 'LemonTree', 'RedFox', 'Ibys'], dtype=object)

df.FoodPreference.unique
<bound method Series.unique of 0          veg
1      Non-Veg
2          Veg
3          Veg
4    Vegetarian
5      Non-Veg
6    Vegetarian
7          Veg
8      Non-Veg
9      non-Veg
Name: FoodPreference, dtype: object>

df.FoodPreference.replace(['Vegetarian', 'veg'], 'Veg', inplace=True)
df.FoodPreference.replace(['non-Veg'], 'Non-Veg', inplace=True)

```

## Result:

Thus the process of missing data values handling is carried out using pandas library in Python.

### **3.B Data Preprocessig In Data Science**

#### **Aim:**

To understand the data preprocessing in Data Science and understand the importance of data preprocessing in data science.

#### **Algorithm:**

*Step 1: Data Cleaning*

*Step 2: Data Transformation*

*Step 3: Feature Engineering*

*Step 4: Data Scaling and Encoding*

#### **Code With Output:**

```
import numpy as np
import pandas as pd
df=pd.read_csv(r"D:\REC 2nd Year\Data Science\Data Sets\
Pre_Process_Data.csv")
df
```

	Country	Age	Salary	Purchased
0	France	44.0	72000.0	No
1	Spain	27.0	48000.0	Yes
2	Germany	30.0	54000.0	No
3	Spain	38.0	61000.0	No
4	Germany	40.0	NaN	Yes
5	France	35.0	58000.0	Yes
6	Spain	NaN	52000.0	No
7	France	48.0	79000.0	Yes
8	Germany	50.0	83000.0	No
9	France	37.0	67000.0	Yes

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Country     10 non-null     object
1   Age         9 non-null      float64
2   Salary      9 non-null      float64
3   Purchased   10 non-null     object
dtypes: float64(2), object(2)
memory usage: 452.0+ bytes
```

```
df['Country'].mode()

0    France
Name: Country, dtype: object

df.Country.mode()[0]

'France'
```

```
df.Country.fillna(df.Country.mode()[0],inplace=True)
df.Age.fillna(df.Age.median(),inplace=True)
df.Salary.fillna(round(df.Salary.mean()),inplace=True)
df
```

```
df.Salary.fillna(round(df.Salary.mean()),inplace=True)
```

	Country	Age	Salary	Purchased
0	France	44.0	72000.0	No
1	Spain	27.0	48000.0	Yes
2	Germany	30.0	54000.0	No
3	Spain	38.0	61000.0	No
4	Germany	40.0	63778.0	Yes
5	France	35.0	58000.0	Yes
6	Spain	38.0	52000.0	No
7	France	48.0	79000.0	Yes
8	Germany	50.0	83000.0	No
9	France	37.0	67000.0	Yes



```
pd.get_dummies(df.Country)
```

	France	Germany	Spain
0	True	False	False
1	False	False	True
2	False	True	False
3	False	False	True
4	False	True	False
5	True	False	False
6	False	False	True
7	True	False	False
8	False	True	False
9	True	False	False

```
updated_dataset=pd.concat([pd.get_dummies(df.Country),df.iloc[:,  
[1,2,3]]],axis=1)  
updated_dataset
```

	France	Germany	Spain	Age	Salary	Purchased
0	True	False	False	44.0	72000.0	No
1	False	False	True	27.0	48000.0	Yes
2	False	True	False	30.0	54000.0	No
3	False	False	True	38.0	61000.0	No
4	False	True	False	40.0	63778.0	Yes
5	True	False	False	35.0	58000.0	Yes
6	False	False	True	38.0	52000.0	No
7	True	False	False	48.0	79000.0	Yes
8	False	True	False	50.0	83000.0	No
9	True	False	False	37.0	67000.0	Yes

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 10 entries, 0 to 9  
Data columns (total 4 columns):  
#   Column      Non-Null Count  Dtype  
---  ---  
0   Country     10 non-null     object  
1   Age         10 non-null     float64  
2   Salary      10 non-null     float64  
3   Purchased   10 non-null     object  
dtypes: float64(2), object(2)  
memory usage: 452.0+ bytes
```

```
updated_dataset.Purchased.replace(['No','Yes'],[0,1],inplace=True)  
updated_dataset
```

	France	Germany	Spain	Age	Salary	Purchased
0	True	False	False	44.0	72000.0	0
1	False	False	True	27.0	48000.0	1
2	False	True	False	30.0	54000.0	0
3	False	False	True	38.0	61000.0	0
4	False	True	False	40.0	63778.0	1
5	True	False	False	35.0	58000.0	1
6	False	False	True	38.0	52000.0	0
7	True	False	False	48.0	79000.0	1
8	False	True	False	50.0	83000.0	0
9	True	False	False	37.0	67000.0	1

### **3.C. Create And Save a CSV file Using Pandas**

#### **Aim:**

To create a CSV File and save the file using Pandas Library in python

#### **Algorithm:**

1. Import Pandas
2. Prepare Data
3. Create a Dataframe
4. Save to CSV File

#### **Description:**

Follow Steps to create and save a file a CSV File using Pandas Library

#### **Code With Output:**

```
import pandas as pd
import random
from datetime import datetime, timedelta
```

```

# Step 1: Define columns
columns = [
    "Member ID", "Name", "Age", "Gender",
    "Plan_Type", "Join_Date", "Expiry_Date", "Trainer_Name"
]

# Step 2: Sample data lists
names = [
    "Aarav", "Ananya", "Vihaan", "Ishita", "Advait",
    "Meera", "Rohan", "Priya", "Kabir", "Simran",
    "Arjun", "Neha", "Yash", "Sanya", "Kunal",
    "Ritika", "Omkar", "Ira", "Aditya", "Tanya",
    "Siddharth", "Pooja", "Manav", "Shruti", "Aditi"
]
genders = ["Male", "Female"]
plans = ["Monthly", "Quarterly", "Yearly"]
trainers = ["Raj", "Priya", "Amit", "Sonal", "Vikram"]

# Step 3: Generate dataset
data = []
start_date = datetime(2025, 1, 1)

for i in range(25):
    member_id = f"M{100+i}"
    name = names[i]
    age = random.randint(18, 50)
    gender = random.choice(genders)
    plan = random.choice(plans)

    join_date = start_date + timedelta(days=random.randint(0, 60))

    if plan == "Monthly":
        expiry_date = join_date + timedelta(days=30)
    elif plan == "Quarterly":
        expiry_date = join_date + timedelta(days=90)
    else:
        expiry_date = join_date + timedelta(days=365)

    trainer = random.choice(trainers)

    data.append([
        member_id, name, age, gender, plan,
        join_date.strftime("%Y-%m-%d"),
        expiry_date.strftime("%Y-%m-%d"),
        trainer
    ])

```

```

# Step 4: Create DataFrame
df = pd.DataFrame(data, columns=columns)

```

```
# Step 5: Save to CSV
df.to_csv("gym_members.csv", index=False)

print(" gym_members.csv file created with 25 entries.")
print(df.head())
```

gym\_members.csv file created with 25 entries.

	Member ID	Name	Age	Gender	Plan Type	Join Date	Expiry Date	\
0	M100	Aarav	41	Female	Yearly	2025-02-02	2026-02-02	
1	M101	Ananya	41	Female	Yearly	2025-02-22	2026-02-22	
2	M102	Vihaan	49	Male	Quarterly	2025-02-22	2025-05-23	
3	M103	Ishita	22	Female	Yearly	2025-01-12	2026-01-12	
4	M104	Advait	35	Male	Yearly	2025-02-23	2026-02-23	

  

	Trainer Name
0	Sonal
1	Sonal
2	Raj
3	Priya
4	Raj

## Result:

Thus a dataset is created in CSV format and saved using Pandas Library