**Day-16(2311cs020140)**

**Cleaning and Preparing Healthcare Data for Analysis**

**Objective:**

To clean a real-world healthcare dataset by handling inconsistencies, duplicates, and missing values.

**Instructions:**

1. **Load the Dataset:**
   * Read the healthcare dataset into a Pandas DataFrame.
2. **Handle Missing Data:**
   * Identify missing values in patient demographics (age, gender, blood pressure, etc.).
   * Apply appropriate imputation methods.
3. **Detect and Handle Duplicates:**
   * Identify duplicate records using duplicated().
   * Remove or merge duplicates as necessary.
4. **Detect and Handle Outliers:**
   * Use boxplots to identify extreme values.
   * Apply transformations or capping techniques to handle outliers.
5. **Standardize and Normalize Data:**
   * Convert categorical variables into numerical representations.
   * Scale numerical variables using Min-Max Scaling or Standard Scaling.
6. **Data Validation:**
   * Ensure no missing values or duplicates remain.
   * Check data types and correct inconsistencies.
7. **Final Data Export:**
   * Save the cleaned dataset as a CSV file for further analysis.

Program:

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.preprocessing

import StandardScaler, MinMaxScaler

from sklearn.impute import SimpleImputer

df = pd.read\_csv('healthcare.csv')

df.head()

missing\_data = df.isna().sum()

missing\_percentage = (missing\_data / len(df)) \* 100

print("Missing Data Count:\n", missing\_data)

print("Missing Data Percentage:\n", missing\_percentage)

num\_columns = df.select\_dtypes(include=[np.number]).columns

cat\_columns = df.select\_dtypes(include=[np.object]).columns

imputer = SimpleImputer(strategy='median')

df[num\_columns] = imputer.fit\_transform(df[num\_columns])

imputer = SimpleImputer(strategy='most\_frequent')

df[cat\_columns] = imputer.fit\_transform(df[cat\_columns])

duplicates = df[df.duplicated()]

print(f"Number of Duplicate Records: {duplicates.shape[0]}")

df = df.drop\_duplicates()

plt.figure(figsize=(12, 8))

sns.boxplot(data=df[num\_columns])

plt.title('Boxplot to Detect Outliers')

plt.show()

for col in num\_columns:

Q1 = df[col].quantile(0.25)

Q3 = df[col].quantile(0.75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

df[col] = np.clip(df[col], lower\_bound, upper\_bound)

df = pd.get\_dummies(df, drop\_first=True)

scaler = MinMaxScaler()

df[num\_columns] = scaler.fit\_transform(df[num\_columns])

assert df.isna().sum().sum() == 0, "There are still missing values in the dataset."

assert df.duplicated().sum() == 0, "There are still duplicate records in the dataset."

print(df.dtypes)

df.to\_csv('cleaned\_healthcare\_data.csv', index=False)

Output:

1.

patient\_id age gender blood\_pressure cholesterol smoking\_status

0 1 34.0 Male 120 200 Smoker

1 2 50.0 Female 130 210 Non-smoker

2 3 45.0 Male 125 215 Non-smoker

3 4 60.0 Female 140 220 Smoker

4 5 55.0 Male 135 225 Non-smoker

2.

Missing Data Count:

patient\_id 0

age 2

gender 0

blood\_pressure 5

cholesterol 3

smoking\_status 0

dtype: int64

Missing Data Percentage:

patient\_id 0.0

age 2.5

gender 0.0

blood\_pressure 6.25

cholesterol 3.75

smoking\_status 0.0

dtype: float64

3.

Number of Duplicate Records: 0

4.

df[col] = np.clip(df[col], lower\_bound, upper\_bound)

5.

patient\_id age blood\_pressure cholesterol gender\_Male smoking\_status\_Smoker

0 1 34.0 120 200 1 1

1 2 50.0 130 210 0 0

2 3 45.0 125 215 1 0

3 4 60.0 140 220 0 1

4 5 55.0 135 225 1 0

6.

patient\_id int64

age float64

blood\_pressure float64

cholesterol float64

gender\_Male int64

smoking\_status\_Smoker int64

dtype: object