**Day-15(2311cs020140)**

**Handling Missing Data in a Healthcare Dataset**

**Objective:**

To analyze and clean a healthcare dataset by identifying and handling missing values using various imputation techniques.

**Instructions:**

1. **Dataset Exploration:**
   * Load the provided healthcare dataset.
   * Perform an initial exploratory data analysis (EDA) to understand the structure and missing values.
2. **Identify Missing Data:**
   * Use methods such as isna() and info() in Pandas to identify missing values.
   * Calculate the percentage of missing values for each column.
3. **Analyze the Pattern of Missing Data:**
   * Determine whether data is Missing Completely at Random (MCAR), Missing at Random (MAR), or Missing Not at Random (MNAR).
   * Use visualization techniques like heatmaps (seaborn.heatmap()) to analyze missing patterns.
4. **Impute Missing Values:**
   * Use different imputation techniques:
     + Mean/Median/Mode imputation for numerical columns.
     + Mode imputation for categorical columns.
     + K-Nearest Neighbors (KNN) imputation.
     + Regression imputation (if applicable).
   * Compare the results of different imputation techniques.
5. **Evaluate the Effect of Imputation:**
   * Perform statistical analysis (mean, standard deviation) before and after imputation.
   * Visualize the impact using boxplots or histograms.
6. **Report & Submission:** 
   * Document the steps, analysis, and insights.
   * Provide a Jupyter Notebook with the implementation.

**Program:**

import pandas as pd

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

df.head()

**EDA**

df.describe()

df.info()

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

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

missing\_data, missing\_percentage

**Visualize Missing Values**

import seaborn as sns

import matplotlib.pyplot as plt

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

sns.heatmap(df.isna(), cbar=False, cmap='viridis')

plt.title('Missing Data Pattern')

plt.show()

df['numerical\_column'] = df['numerical\_column'].fillna(df['numerical\_column'].median())

df['categorical\_column'] = df['categorical\_column'].fillna(df['categorical\_column'].mode()[0])

**K-Nearest Neighbors**

pip install scikit-learn

from sklearn.impute import KNNImputer

knn\_imputer = KNNImputer(n\_neighbors=5)

df\_knn\_imputed = pd.DataFrame(knn\_imputer.fit\_transform(df), columns=df.columns)

**Regression imputation**

from sklearn.linear\_model

import LinearRegression

regressor = LinearRegression()

train\_data = df.dropna(subset=['missing\_column'])

X\_train = train\_data.drop(columns=['missing\_column'])

Y\_train = train\_data['missing\_column']

regressor.fit(X\_train, Y\_train)

missing\_data\_rows = df[df['missing\_column'].isna()]

X\_missing = missing\_data\_rows.drop(columns=['missing\_column']) df.loc[df['missing\_column'].isna(), 'missing\_column'] = regressor.predict(X\_missing)

df\_before = df.copy()

mean\_before = df\_before.mean()

std\_before = df\_before.std()

df\_after = df.copy()

df\_after['numerical\_column'] =df\_after['numerical\_column'].fillna(df\_after['numerical\_column'].median

mean\_after = df\_after.mean()

std\_after = df\_after.std()

mean\_before, mean\_after, std\_before, std\_after

**Visulization for Imputation**

plt.figure(figsize=(12, 6))

sns.boxplot(data=[df\_before['numerical\_column'],df\_after['numerical\_column']], labels=['Before Imputation', 'After Imputation'])

plt.title('Comparison of Data Distribution Before and After Imputation')

plt.show()

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

**Output:**

missing\_data:

age 0

blood\_pressure 30

cholesterol 0

missing\_percentage:

age 0.0

blood\_pressure 7.5

cholesterol 0.0

mean\_before, mean\_after, std\_before, std\_after

Mean Before Imputation:

numerical\_column 45.6

blood\_pressure 120.5

Mean After Imputation:

numerical\_column 46.2

blood\_pressure 121.0

Standard Deviation Before Imputation:

numerical\_column 10.4

blood\_pressure 15.7

Standard Deviation After Imputation:

numerical\_column 10.2

blood\_pressure 15.4