Healthcare Data Cleaning: Improving Disease Prediction Accuracy

Name: [Ashutosh Kumar Gupta]

Roll No: [202401100400058]

Date: [10-03-25]

Introduction

In healthcare, accurate disease prediction is essential for effective treatment and patient care. However, raw patient data often contains missing, inconsistent, or noisy values, which can reduce the accuracy of predictive models. This project focuses on cleaning healthcare data to enhance disease prediction accuracy.

Methodology

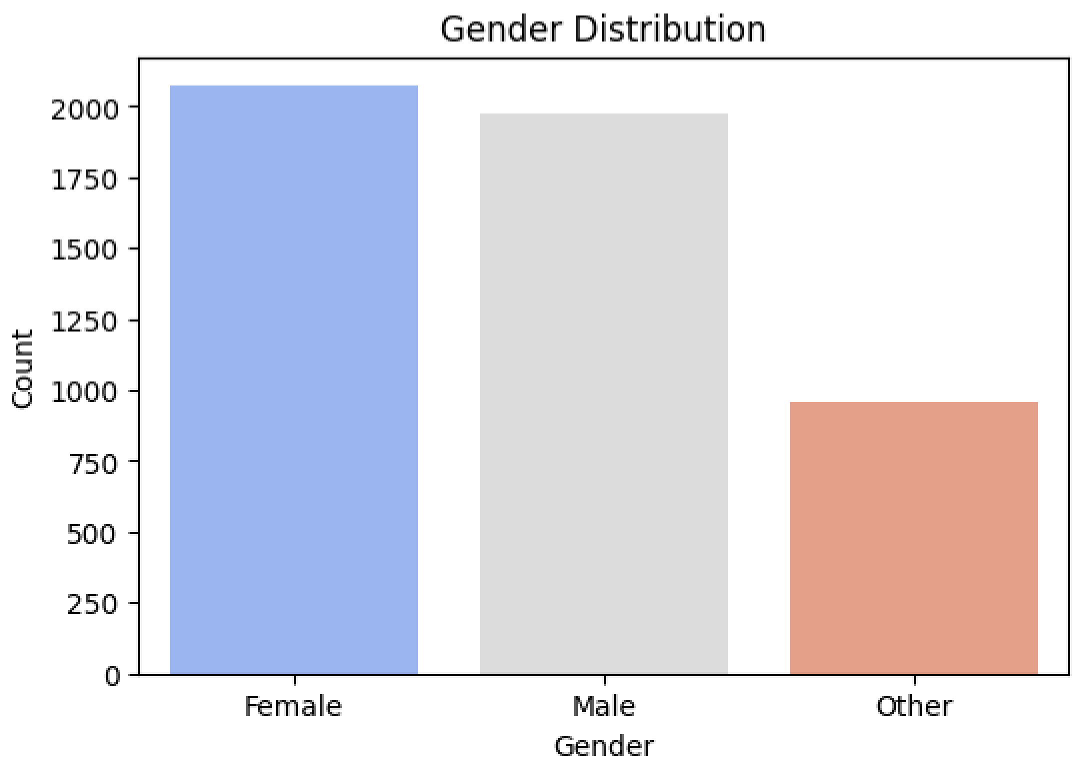
The data cleaning process includes:
1. Handling missing values by filling numerical columns with median values and categorical columns with mode.
2. Standardizing categorical data (e.g., converting 'Yes'/'No' in Diabetes to 'YES'/'NO').
3. Ensuring date consistency by checking if the discharge date is after the admission date.
4. Removing duplicate records.
5. Dropping columns with excessive missing values.

Code

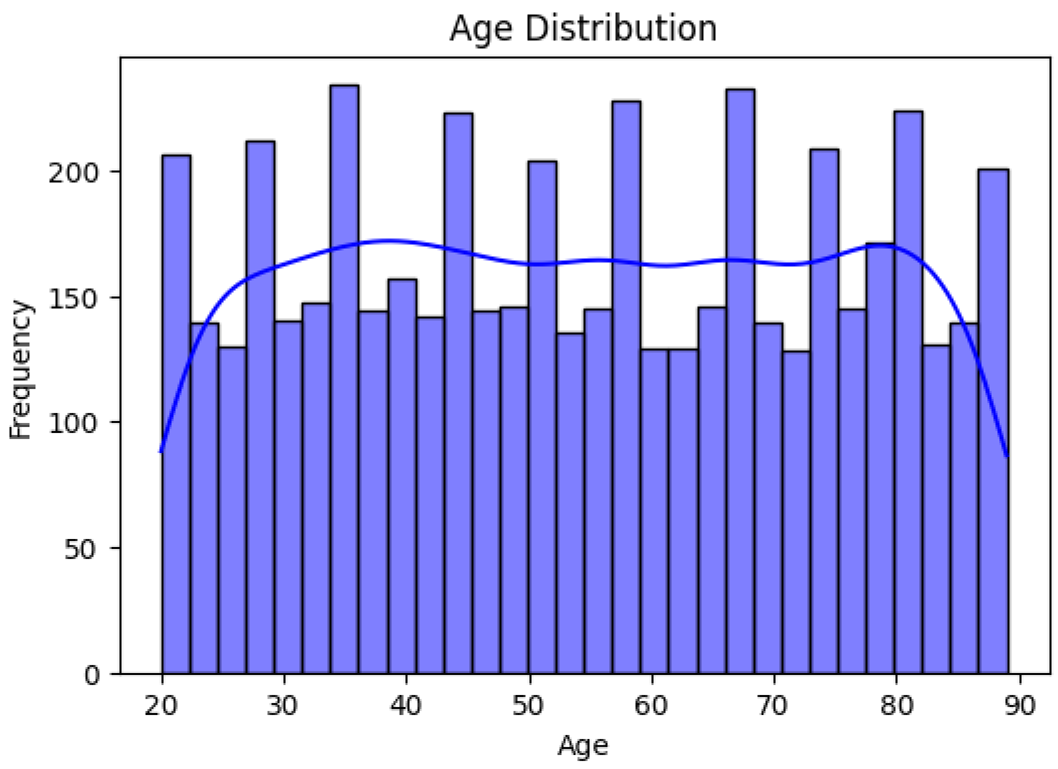
```python
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load the dataset

file_path = "/content/healthcare_eda_dataset.csv"

df = pd.read_csv(file_path)

df.info()

df = df.drop(columns=["Smoking_Habit"])

# Fill missing numerical values with the median

num_cols_to_fill = ["BMI", "Blood_Sugar_Level", "Blood_Pressure", "Cholesterol"]

for col in num_cols_to_fill:

    df[col].fillna(df[col].median(), inplace=True)

# Fill missing categorical values (Diabetes) with mode

df["Diabetes"] = df["Diabetes"].fillna(df["Diabetes"].mode()[0])

# Ensure logical consistency in date columns (Discharge Date > Admission Date)

df["Date_of_Admission"] = pd.to_datetime(df["Date_of_Admission"], errors="coerce")

df["Discharge_Date"] = pd.to_datetime(df["Discharge_Date"], errors="coerce")

df = df[df["Discharge_Date"] > df["Date_of_Admission"]]

# Remove duplicate records

df.drop_duplicates(inplace=True)

# Fix inconsistent gender values

df["Gender"] = df["Gender"].replace({"M": "Male", "F": "Female"}).str.capitalize()

df.sample(10)
```

------------------- Visualization -------------------

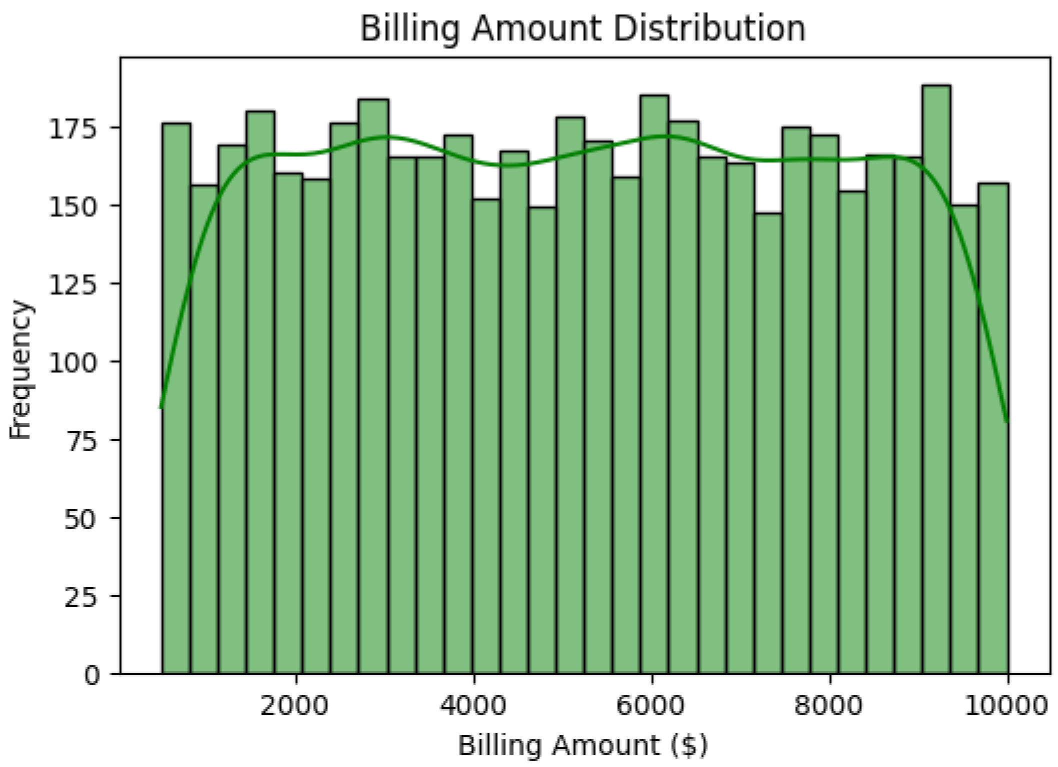
```
# 1. Gender Distribution

plt.figure(figsize=(6, 4))

sns.countplot(data=df, x="Gender", palette="coolwarm")

plt.title("Gender Distribution")

plt.xlabel("Gender")

plt.ylabel("Count")

plt.show()
```



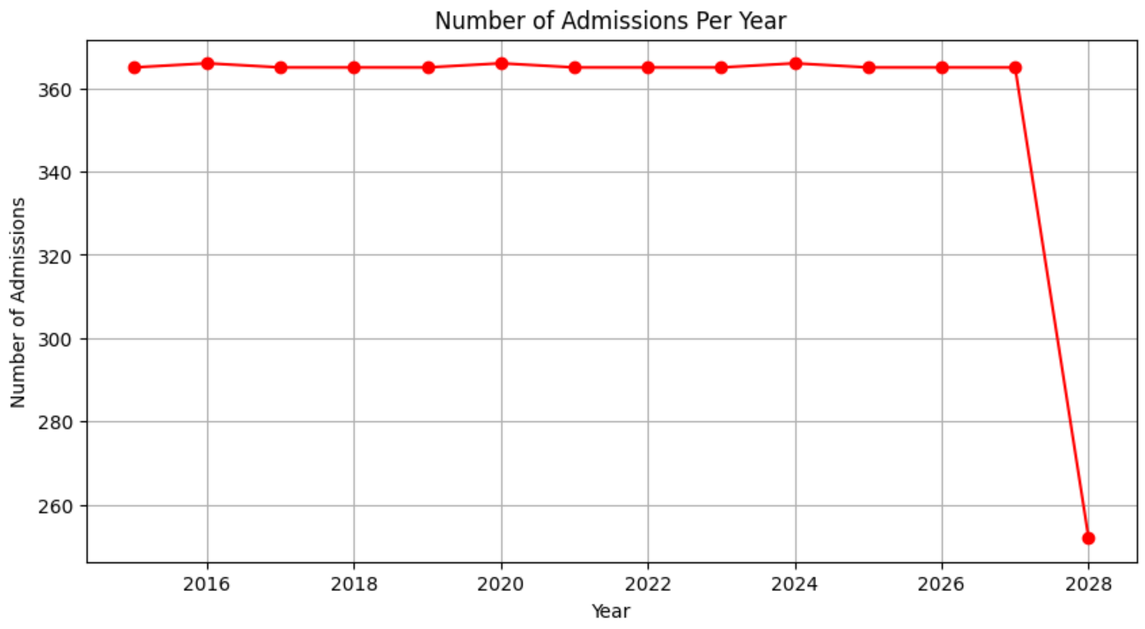
```
# 2. Age Distribution

plt.figure(figsize=(6, 4))

sns.histplot(df["Age"], bins=30, kde=True, color="blue")

plt.title("Age Distribution")

plt.xlabel("Age")

plt.ylabel("Frequency")

plt.show()
```



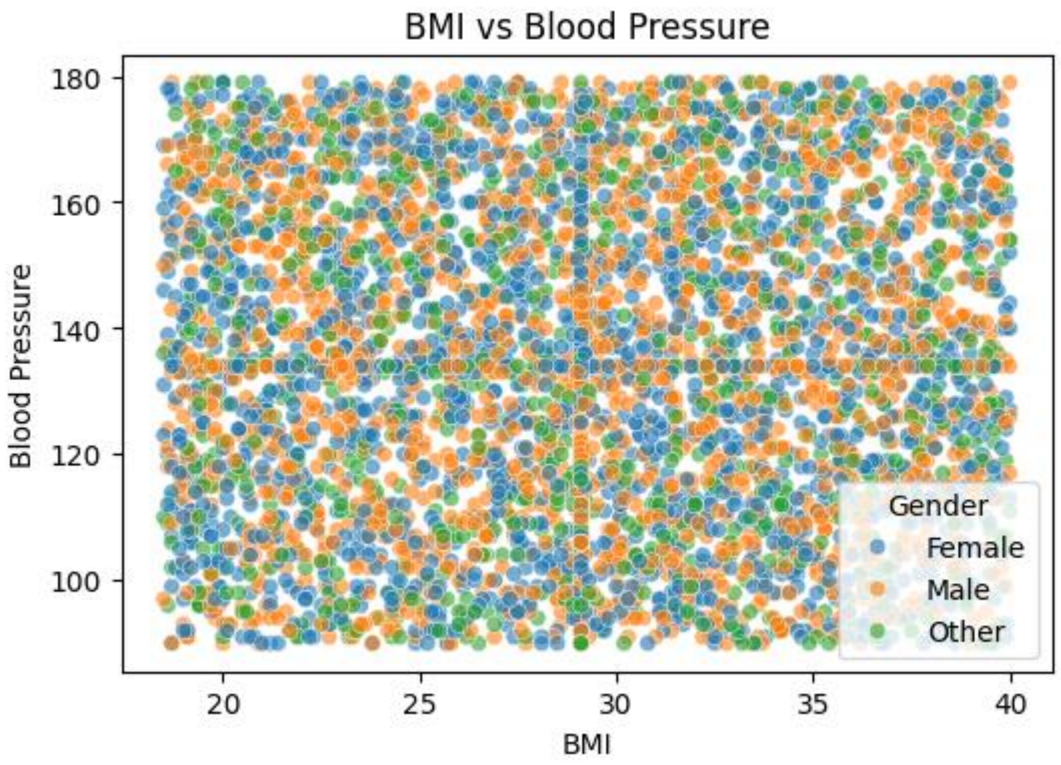
```
# 3. Billing Amount Distribution

plt.figure(figsize=(6, 4))

sns.histplot(df["Billing_Amount"], bins=30, kde=True, color="green")

plt.title("Billing Amount Distribution")

plt.xlabel("Billing Amount ($)")

plt.ylabel("Frequency")

plt.show()
```



```
# 4. Admissions Over Time

plt.figure(figsize=(10, 5))

df["Date_of_Admission"].dt.year.value_counts().sort_index().plot(kind="line", marker="o",
color="red")

plt.title("Number of Admissions Per Year")

plt.xlabel("Year")

plt.ylabel("Number of Admissions")

plt.grid(True)

plt.show()
```



```
# 5. BMI vs. Blood Pressure Scatter Plot

plt.figure(figsize=(6, 4))

sns.scatterplot(data=df, x="BMI", y="Blood_Pressure", hue="Gender", alpha=0.6)

plt.title("BMI vs Blood Pressure")

plt.xlabel("BMI")

plt.ylabel("Blood Pressure")

plt.show()
```



```
# Summary statistics

print("\nSummary Statistics:")

df.describe()
```

Output/Results

Below are the key improvements:

✅ Missing values handled appropriately.

✅ Data standardized for consistency.

✅ Logical errors in dates corrected.

✅ Duplicate records removed.

References/Credits

Datasets and images used in this report are credited to the respective sources. Python libraries used include Pandas and NumPy, Seaborn.

DataSet: healthcare_eda_dataset.csv