Healthcare Provider Fraud Detection Data Cleaning Process

Author: Artur Melnyk
Date: 2024-08-01

Table of Contents

---

1. Introduction

This document provides a comprehensive guide to the data cleaning process undertaken for the Healthcare Provider Fraud Detection dataset. The goal of this process was to ensure that the data was clean, consistent, and ready for further analysis or modeling.

---

2. Loading Data

Objective:
To load the datasets into pandas DataFrames to facilitate subsequent data cleaning and processing steps.

File Paths:

- **Training Data:**
    - Train_Beneficiarydata-1542865627584.csv
    - Train_Inpatientdata-1542865627584.csv
    - Train_Outpatientdata-1542865627584.csv
    - Train-1542865627584.csv

- **Test Data:**
    - Test_Beneficiarydata-1542969243754.csv
    - Test_Inpatientdata-1542969243754.csv
    - Test_Outpatientdata-1542969243754.csv
    - Test-1542969243754.csv

Process:
The datasets were loaded into pandas DataFrames using the pd.read_csv() function, ensuring they were ready for the cleaning process.

```
# Load training datasets
train_beneficiary = pd.read_csv(train_beneficiary_path)
train_inpatient = pd.read_csv(train_inpatient_path)
train_outpatient = pd.read_csv(train_outpatient_path)
train = pd.read_csv(train_path)

# Load test datasets
test_beneficiary = pd.read_csv(test_beneficiary_path)
test_inpatient = pd.read_csv(test_inpatient_path)
test_outpatient = pd.read_csv(test_outpatient_path)
test = pd.read_csv(test_path)
```

---

3. Handling Missing Values

Objective:
To address missing values in the datasets to ensure data completeness.

Process:

- **Beneficiary Data:**
    - **Date of Death (DOD):** Missing values were filled with a placeholder date ('2100-01-01'). This was necessary to differentiate between missing values and actual dates.
    - **Numerical Columns:** Missing values in numerical columns were filled with the median value of each column.

- **Categorical Columns:** Missing values in categorical columns were filled with the mode (most frequent value) of each column.

Challenges Addressed:

- Placeholder dates needed to be handled carefully to avoid affecting future analyses.

```
def data_cleaning_beneficiary(df):
    # Fill missing values in the 'DOD' column with a placeholder date
    df['DOD'].fillna('2100-01-01', inplace=True)

    # Fill missing values for numerical columns with median
    numerical_cols_beneficiary = df.select_dtypes(include=['float64', 'int64']).columns
    df[numerical_cols_beneficiary] =
df[numerical_cols_beneficiary].fillna(df[numerical_cols_beneficiary].median())
    # Fill missing values for categorical columns with mode
    categorical_cols_beneficiary = df.select_dtypes(include=['object']).columns
    df[categorical_cols_beneficiary] =
df[categorical_cols_beneficiary].fillna(df[categorical_cols_beneficiary].mode().iloc[0])

    # Convert date columns to datetime type
    df['DOB'] = pd.to_datetime(df['DOB'])
    df['DOD'] = pd.to_datetime(df['DOD'])

    # Ensure categorical columns are of type 'category'
    categorical_cols = ['Gender', 'Race', 'RenalDiseaseIndicator', 'State', 'County']
    df[categorical_cols] = df[categorical_cols].astype('category')

    return df
```

---

4. Handling Outliers

Objective:
To identify and handle outliers in the numerical data to ensure robust analysis.

Process:

- **Outlier Removal:** Used the Interquartile Range (IQR) method to remove outliers from numerical columns. The IQR method helps in identifying outliers by calculating the range within which most data points lie and removing data points outside this range.

Challenges Addressed:

- Ensured that significant data points were not removed while handling outliers.

```
def handle_outliers_iqr(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    return df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]
```

---

5. General Data Cleaning

Objective:
To apply outlier handling to numerical columns in the given DataFrame.

Process:

- **Outlier Handling:** Applied the IQR method to all numerical columns in the DataFrame to remove outliers and ensure data integrity.

Challenges Addressed:

- Ensured that the general cleaning process effectively handled outliers in various datasets.

```
def data_cleaning_general(df):
    numerical_columns = df.select_dtypes(include=['float64', 'int64']).columns
    for col in numerical_columns:
```

```
    df = handle_outliers_iqr(df, col)
  return df
```

---

6. Data Type Conversion

Objective:
To ensure all columns have appropriate data types for accurate analysis.

Process:

- **Date Columns:** Converted specified columns to datetime format to enable proper date-time operations.
- **Categorical Columns:** Converted specified columns to the 'category' data type to optimize memory usage and ensure correct handling of categorical data.

Challenges Addressed:

- Ensured correct conversion of date and categorical columns while maintaining consistency.

```
def convert_date_columns(df, date_columns):
  for col in date_columns:
    df[col] = pd.to_datetime(df[col], errors='coerce')
  return df


def convert_to_category(df, categorical_columns):
  df[categorical_columns] = df[categorical_columns].astype('category')
  return df
```

---

7. Applying Data Cleaning

Objective:
To apply the cleaning functions to all relevant datasets to ensure data quality.

Process:

- **Beneficiary Data:** Applied specific cleaning procedures to handle missing values, outliers, and data type conversions.
- **Inpatient and Outpatient Data:** Applied general cleaning procedures to handle missing values, outliers, and data type conversions.

- **Date Columns Conversion:** Ensured date columns were converted to the correct data type in inpatient and outpatient datasets.
- **Categorical Columns Conversion:** Ensured categorical columns were converted to the correct data type in inpatient and outpatient datasets.

```
# Apply data cleaning to beneficiary datasets
train_beneficiary = data_cleaning_beneficiary(train_beneficiary)
test_beneficiary = data_cleaning_beneficiary(test_beneficiary)

# Apply data cleaning to inpatient and outpatient datasets
train_inpatient = data_cleaning_general(train_inpatient)
test_inpatient = data_cleaning_general(test_inpatient)

train_outpatient = data_cleaning_general(train_outpatient)
test_outpatient = data_cleaning_general(test_outpatient)
# Convert date columns for inpatient data
date_columns_inpatient = ['ClaimStartDt', 'ClaimEndDt', 'AdmissionDt', 'DischargeDt']
train_inpatient = convert_date_columns(train_inpatient, date_columns_inpatient)
test_inpatient = convert_date_columns(test_inpatient, date_columns_inpatient)

# Convert date columns for outpatient data
date_columns_outpatient = ['ClaimStartDt', 'ClaimEndDt']
train_outpatient = convert_date_columns(train_outpatient, date_columns_outpatient)
test_outpatient = convert_date_columns(test_outpatient, date_columns_outpatient)

# Convert categorical columns for inpatient data
categorical_columns_inpatient = ['AttendingPhysician', 'OperatingPhysician', 'OtherPhysician',
'ClmAdmitDiagnosisCode', 'DiagnosisGroupCode'] + [f'ClmDiagnosisCode_{i}' for i in range(1, 11)] +
[f'ClmProcedureCode_{i}' for i in range(1, 7)]
train_inpatient = convert_to_category(train_inpatient, categorical_columns_inpatient)
```

```
test_inpatient = convert_to_category(test_inpatient, categorical_columns_inpatient)


# Convert categorical columns for outpatient data
categorical_columns_outpatient = ['AttendingPhysician', 'OperatingPhysician', 'OtherPhysician', 'ClmAdmitDiagnosisCode'] + [f'ClmDiagnosisCode_{i}' for i in range(1, 11)] + [f'ClmProcedureCode_{i}' for i in range(1, 7)]
train_outpatient = convert_to_category(train_outpatient, categorical_columns_outpatient)
test_outpatient = convert_to_category(test_outpatient, categorical_columns_outpatient)
```

---

8. Verification

Objective:
To verify that the data cleaning process was successful and that the datasets were free of issues.

Process:

- **Missing Values Check:** Checked for any remaining missing values in the datasets to ensure completeness.
- **Data Type Check:** Verified the data types of all columns to ensure they were correctly set for further analysis.

```
# Verify Train_Beneficiarydata
print("Missing values in Train_Beneficiarydata after all steps:")
print(train_beneficiary.isnull().sum())
print("Data types in Train_Beneficiarydata after all steps:")
print(train_beneficiary.dtypes)


# Verify Test_Beneficiarydata
print("Missing values in Test_Beneficiarydata after all steps:")
print(test_beneficiary.isnull().sum())
print("Data types in Test_Beneficiarydata after all steps:")
print(test_beneficiary.dtypes)
```

```
# Verify Train_Inpatientdata
print("Missing values in Train_Inpatientdata after all steps:")
print(train_inpatient.isnull().sum())
print("Data types in Train_Inpatientdata after all steps:")
print(train_inpatient.dtypes)

# Verify Test_Inpatientdata
print("Missing values in Test_Inpatientdata after all steps:")
print(test_inpatient.isnull().sum())
print("Data types in Test_Inpatientdata after all steps:")
print(test_inpatient.dtypes)

# Verify Train_Outpatientdata
print("Missing values in Train_Outpatientdata after all steps:")
print(train_outpatient.isnull().sum())
print("Data types in Train_Outpatientdata after all steps:")
print(train_outpatient.dtypes)

# Verify Test_Outpatientdata
print("Missing values in Test_Outpatientdata after all steps:")
print(test_outpatient.isnull().sum())
print("Data types in Test_Outpatientdata after all steps:")
print(test_outpatient.dtypes)

# Verify Train
print("Missing values in Train after all steps:")
print(train.isnull().sum())
print("Data types in Train after all steps:")
print(train.dtypes)
```

```
# Verify Test
print("Missing values in Test after all steps:")
print(test.isnull().sum())
print("Data types in Test after all steps:")
print(test.dtypes)
```

---

9. Recommendations for Further Improvements

Handling Placeholder Dates:

- Replace placeholder dates in the 'DOD' column with NaT (Not a Time) to ensure they do not interfere with analysis.

Creating New Features:

- Create new features such as Age from DOB and DaysAliveAfterClaim from DOD and ClaimEndDt to provide additional insights.

Consistency Check:

- Ensure all BeneID values in inpatient and outpatient datasets are present in the beneficiary dataset to maintain data consistency.

Categorical Data Encoding:

- Use label encoding or one-hot encoding to prepare categorical data for machine learning models.

Removing Unnecessary Columns:

- Continuously evaluate and remove columns that are not useful for analysis to streamline the datasets.

---

10. Conclusion

The data cleaning process was successful in handling missing values, converting data types, handling outliers, ensuring consistency, and preparing the datasets for analysis or modeling. The additional recommendations provide further steps to enhance data quality and readiness for future tasks.

Detailed Process Overview

Loading Data

Datasets were loaded using the pd.read_csv() function, ensuring they were in pandas DataFrames for further processing.

Handling Missing Values

- **DOD Column:** Missing values were filled with a placeholder date ('2100-01-01').
- **Numerical Columns:** Missing values were filled with the median.
- **Categorical Columns:** Missing values were filled with the mode.

Handling Outliers

The Interquartile Range (IQR) method was used to remove outliers from numerical columns.

Data Type Conversion

- **Date Columns:** Converted to datetime format.
- **Categorical Columns:** Converted to the 'category' data type.

Applying Data Cleaning

Cleaning functions were applied to the beneficiary, inpatient, and outpatient datasets. Date and categorical columns were converted to their appropriate types.

Verification

Checks were performed to ensure no missing values remained and that data types were correctly set.

Recommendations

- **Handling Placeholder Dates:** Replace placeholder dates with NaT.
- **Creating New Features:** Calculate Age and DaysAliveAfterClaim.
- **Consistency Check:** Ensure all BeneID values are present across datasets.
- **Categorical Data Encoding:** Use label encoding or one-hot encoding.
- **Removing Unnecessary Columns:** Continuously remove non-essential columns.

This documentation ensures the data cleaning process is well-documented, facilitating a clear understanding of the steps taken and the challenges addressed.