Data Loading and Preprocessing

Loading the healthcare information, which included numerous patient-related characteristics and billing amounts, was the initial stage in the analysis process. We used the pandas library to read the CSV file into a Data Frame after importing the required libraries. Several crucial steps were taken during the preprocessing phase:  
  
Date Conversion: To make it easier to calculate the duration of hospital stays, date columns (such as "Date of Admission" and "Discharge Date") were converted into datetime format.

Length of Stay, a new feature that is determined by subtracting the admission date from the discharge date, was created.

Outlier Removal: To protect the integrity of the data, any records with negative billing amounts were filtered out to address possible outliers.  
Large categorical data (such as physicians, hospitals, and insurance companies) were label-encoded using categorical encoding, whereas smaller categorical features.

Model Training

Training a prediction model was the next step after preprocessing the data. The target variable (y), in this example the billing amount, was isolated from the features (X). To make sure the model could be tested on unseen data, the dataset was divided into training and testing sets in an 80-20 ratio. Due to its ease of use and interpretability, we first used a linear regression model. The training set was used to teach the model the connections between the target variable and the features.

Evaluation using Mean Squared Error (MSE)

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import LabelEncoder

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

# Load the dataset

file\_path = 'healthcare\_dataset.csv'

healthcare\_data = pd.read\_csv(file\_path)

# Convert date columns to datetime format

healthcare\_data['Date of Admission'] = pd.to\_datetime(healthcare\_data['Date of Admission'])

healthcare\_data['Discharge Date'] = pd.to\_datetime(healthcare\_data['Discharge Date'])

# Create a new feature: length of stay (in days)

healthcare\_data['Length of Stay'] = (healthcare\_data['Discharge Date'] - healthcare\_data['Date of Admission']).dt.days

# Drop date columns

healthcare\_data = healthcare\_data.drop(columns=['Date of Admission', 'Discharge Date'])

# Handle outliers: Remove negative billing amounts

healthcare\_data = healthcare\_data[healthcare\_data['Billing Amount'] > 0]

# Label encode large categorical features

label\_enc\_cols = ['Doctor', 'Hospital', 'Insurance Provider']

le = LabelEncoder()

for col in label\_enc\_cols:

    healthcare\_data[col] = le.fit\_transform(healthcare\_data[col])

# Apply one-hot encoding for smaller categorical columns

categorical\_columns = ['Gender', 'Blood Type', 'Medical Condition', 'Admission Type', 'Medication', 'Test Results']

healthcare\_data = pd.get\_dummies(healthcare\_data, columns=categorical\_columns, drop\_first=True)

# Drop irrelevant columns

healthcare\_data = healthcare\_data.drop(columns=['Name', 'Room Number'])

# Separate features (X) and target (y)

X = healthcare\_data.drop(columns=['Billing Amount'])

y = healthcare\_data['Billing Amount']

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Initialize and train a Linear Regression model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Make predictions on the test set

y\_pred = model.predict(X\_test)

# Evaluate the model using Mean Squared Error (MSE)

mse = mean\_squared\_error(y\_test, y\_pred)

print(f"Mean Squared Error (MSE): {mse}")

Mean Squared Error (MSE), which measures the average squared difference between the expected and actual billing amounts, was used to assess the model's performance after it had been trained. A considerable degree of error in the model's predictions was shown by the calculated MSE, which was roughly 203,339,264. This high number indicated that the accuracy of the model might be significantly increased and that additional research was required to improve the strategy.

Reflection on the Problem and Solution

After considering the analysis, a number of important conclusions were drawn:  
  
A high MSE may have resulted from the original method's use of linear regression, which gave a baseline for comprehending the model's predictive power but might not have caught the intricate relationships in the data.  
More advanced models that are better suited to managing non-linear interactions in the dataset, like Random Forest or Gradient Boosting, may be used in future advancements.  
Furthermore, the model's performance may be greatly impacted by expanding the feature set by adding additional features or choosing more pertinent ones.  
A more thorough evaluation of the model's predictive skills might be possible with the use of cross-validation and hyperparameter adjustment.  
Reducing the MSE may also involve addressing outliers and making sure the data is scaled appropriately.