
ALLIANCE UNIVERSITY

*Private University established in Karnataka State by Act No.34 of year 2010*
*Recognized by the University Grants Commission (UGC), New Delhi*

# Project Report

## Bachelor of Computer Applications

Semester II

Introduction to Data Science

By:

Dakshraj Singh Chandawat
Reg No.- 2411021240050
Class- BCA-A

**Department of Computer Applications**

**Alliance University**

**Chandapura-Anekal Main Road, Anekal**

**Bengaluru-562106**

**April 2025**

# Heart Disease Prediction using Logistic Regression

## Introduction:

Heart disease is a major public health concern and one of the leading causes of death globally. Early detection can significantly reduce risks and save lives. In this project, we use a real-world health dataset named heart_2020_cleaned.csv to develop a predictive model that can identify whether a person has heart disease based on various health indicators.

The dataset contains 319,795 entries with 18 columns, including features such as BMI, Smoking, AlcoholDrinking, Stroke, PhysicalHealth, MentalHealth, GeneralHealth, SleepTime, and more. The target variable is HeartDisease, a binary categorical variable indicating whether the individual has heart disease (Yes/No).

The goal of this project is to clean, explore, visualize, and apply machine learning techniques, specifically Logistic Regression, to predict heart disease.

## Objective

The core objective of this project is to leverage data science and machine learning techniques to predict the likelihood of heart disease in individuals based on various health, lifestyle, and demographic parameters. The goal is to build a binary classification model that can accurately distinguish between individuals who are at risk of heart disease and those who are not. This project serves as a practical implementation of logistic regression for binary classification problems, showcasing its utility in solving real-world healthcare challenges.

By employing this predictive model, we aim to:

- Provide early warning indicators based on patient data.
- Help healthcare providers allocate resources more effectively.
- Support preventive care through risk profiling.
- Explore the predictive power of common health factors like BMI, smoking habits, physical activity, and sleep duration.

## Project Goal

The goal of this project is to create a streamlined, efficient, and interpretable model that can classify whether a patient is likely to have heart disease. The emphasis is on interpretability, reliability, and generalization of the model so it could be used as a supportive diagnostic tool.

The specific goals include:

1. Conduct a thorough exploratory data analysis (EDA) to understand feature distributions, correlations, and outliers.
2. Clean and preprocess the dataset including handling of categorical variables, encoding, and standardization.
3. Visualize important patterns and relationships using graphical tools.
4. Train and validate a logistic regression model, suitable for binary outcomes.
5. Evaluate the model's predictive performance using industry-standard metrics such as accuracy, confusion matrix, precision, recall, and F1-score.

**Project Overview**

This project is a comprehensive end-to-end machine learning workflow developed to predict heart disease status using patient health survey data. The dataset used in this study, heart_2020_cleaned.csv, is sourced from a cleaned version of CDC's BRFSS (Behavioral Risk Factor Surveillance System) data.

It includes over 319,000 entries and 18 features such as:

- Behavioral features: Smoking, Alcohol Drinking, Physical Activity, Sleep Time
- Physical/mental health: BMI, PhysicalHealth, MentalHealth
- Chronic conditions: Stroke, Asthma, Kidney Disease, Skin Cancer
- Demographics: Sex, Race, Age Category
- Self-reported metrics: General Health, Diabetes, Difficulty Walking

The pipeline implemented in this project follows key stages of a typical data science lifecycle:

1. Data Collection & Loading: Importing the dataset into a structured format for analysis.
2. Exploratory Data Analysis (EDA): Investigating distributions and relationships between variables.
3. Data Preprocessing: Encoding categorical data, scaling numerical values, and selecting relevant features.
4. Model Building: Implementing logistic regression for binary classification.
5. Model Evaluation: Assessing prediction quality using visual and numeric metrics.
6. Insights & Recommendations: Interpreting the outcomes and suggesting future directions.

This report documents the full journey from raw data to meaningful conclusions, bridging the gap between data science theory and its application in healthcare.

**Code with Explanations:**

**1. Importing Libraries**

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

This step imports the essential libraries required for different stages of the project:

- pandas is used for loading and manipulating structured datasets. It provides efficient data handling operations.
- matplotlib.pyplot is a core visualization library used for generating static plots such as line charts and bar graphs.
- numpy provides support for high-performance numerical computation and array manipulation.
- seaborn is built on top of matplotlib and is tailored for drawing attractive and informative statistical plots.

**2. Loading Data:**

```
df=pd.read_csv(r"C:\Users\mypci\Downloads\heart_2020_cleaned.csv")
df
```

| | HeartDisease | BMI | Smoking | AlcoholDrinking | Stroke | PhysicalHealth | MentalHealth | DiffWalking | Sex | AgeCategory | Race | Diabetic | PhysicalAct |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | No | 16.60 | Yes | No | No | 3.0 | 30.0 | No | Female | 55-59 | White | Yes | |
| 1 | No | 20.34 | No | No | Yes | 0.0 | 0.0 | No | Female | 80 or older | White | No | |
| 2 | No | 26.58 | Yes | No | No | 20.0 | 30.0 | No | Male | 65-69 | White | Yes | |
| 3 | No | 24.21 | No | No | No | 0.0 | 0.0 | No | Female | 75-79 | White | No | |
| 4 | No | 23.71 | No | No | No | 28.0 | 0.0 | Yes | Female | 40-44 | White | No | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 319790 | Yes | 27.41 | Yes | No | No | 7.0 | 0.0 | Yes | Male | 60-64 | Hispanic | Yes | |
| 319791 | No | 29.84 | Yes | No | No | 0.0 | 0.0 | No | Male | 35-39 | Hispanic | No | |
| 319792 | No | 24.24 | No | No | No | 0.0 | 0.0 | No | Female | 45-49 | Hispanic | No | |
| 319793 | No | 32.81 | No | No | No | 0.0 | 0.0 | No | Female | 25-29 | Hispanic | No | |
| 319794 | No | 46.56 | No | No | No | 0.0 | 0.0 | No | Female | 80 or older | Hispanic | No | |

319795 rows × 18 columns

- This line reads a local CSV file and stores its content into a pandas DataFrame called df.
- df.head() displays the first five rows to offer a preview of the dataset, allowing a basic inspection of the feature names and sample values. It helps confirm successful loading.

### 3. Understanding Dataset:

```
df.info()
```

```
df.describe()
```

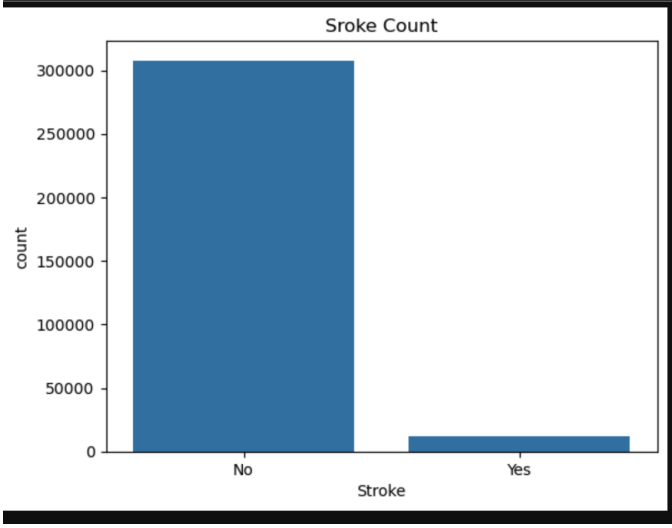
```
df.isnull().sum()
```

- df.info() prints the schema of the dataset, including data types and non-null counts. It is vital for checking if any columns require type conversion or contain missing values.
- df.describe() calculates summary statistics for each numeric column, including count, mean, standard deviation, min, and max. It helps detect outliers or anomalies.
- df.isnull().sum() confirms the absence of null values. In this dataset, the data has already been cleaned.

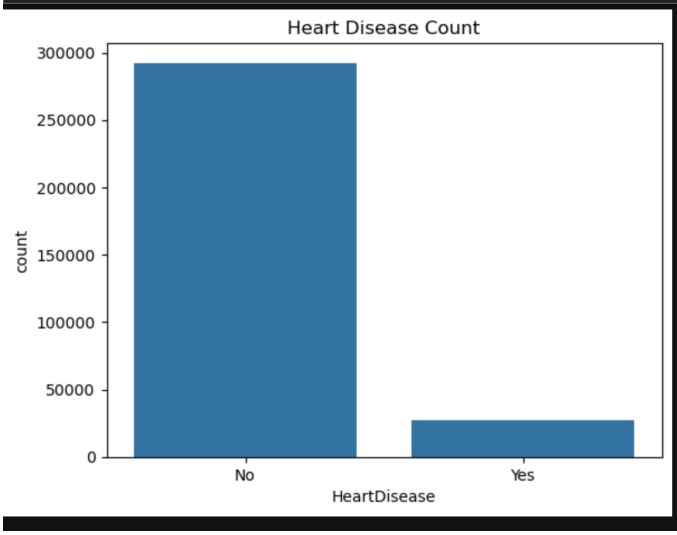
### 4. Visualizations

We visualize the data distribution and relationships:

```
sns.countplot(x='Stroke', data=df)
plt.title('Sroke Count')
plt.show()
```

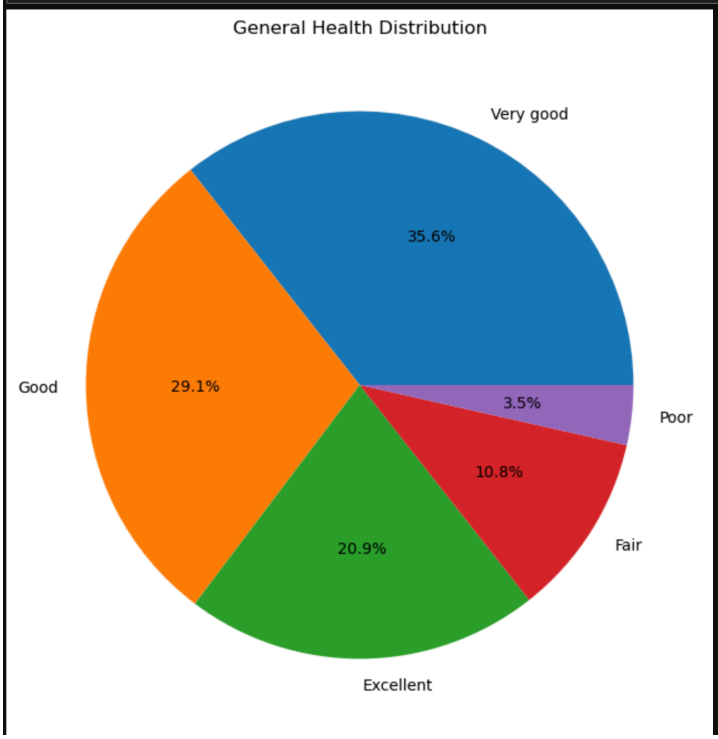



```
sns.countplot(x='HeartDisease', data=df)
plt.title('Heart Disease Count')
plt.show()
```



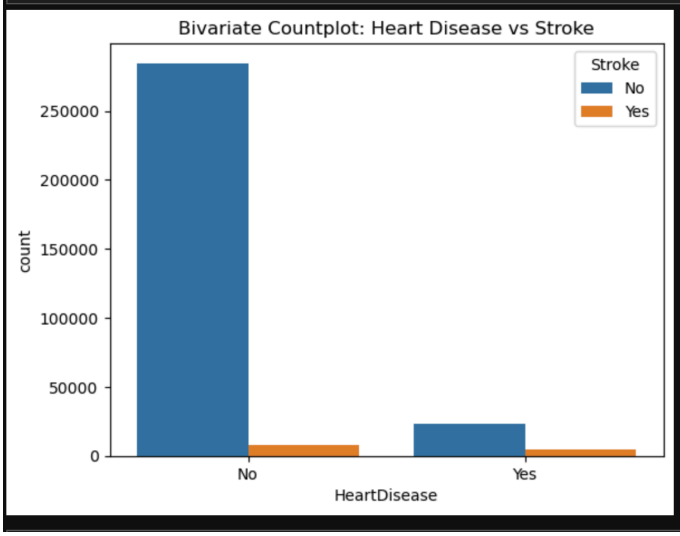

- These count plots show the distribution of binary variables such as Stroke and HeartDisease.
- It becomes immediately clear if the dataset is imbalanced. For example, if most individuals do not have heart disease, this could bias the model.

```
gen_health_counts = df['GenHealth'].value_counts()
gen_health_counts.plot(kind='pie', autopct='%1.1f%%', figsize=(8, 8), title='General Health Distribution')
plt.ylabel('')  # Remove the y-axis label for better visualization
plt.show()
```




- This pie chart displays the percentage breakdown of how individuals rate their general health.
- It helps visualize subjective self-perceptions and potentially correlate them with actual health risks.

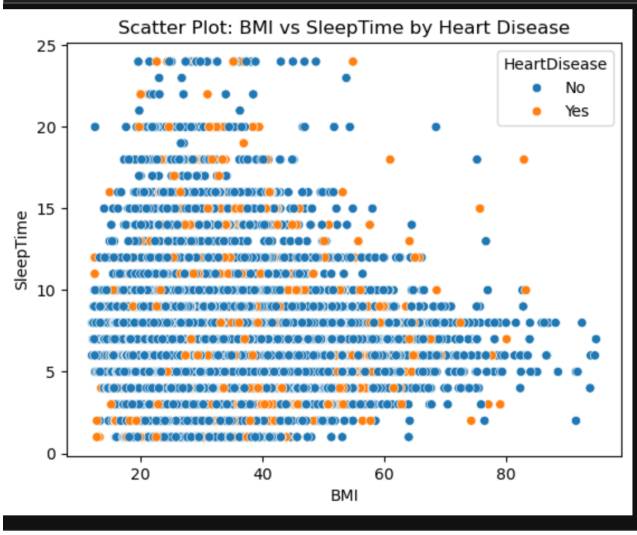
```
sns.countplot(x='HeartDisease', hue='Stroke', data=df)
plt.title('Bivariate Countplot: Heart Disease vs Stroke')
plt.show()
```




These plots explore relationships between variables. For example, whether stroke history is related to heart disease and how BMI and sleep affect heart disease occurrence.

- `hue`: Adds a second category for comparison.
- `scatterplot()`: Shows how two numerical features relate.

```
sns.scatterplot(x='BMI', y='SleepTime', hue='HeartDisease', data=df)
plt.title('Scatter Plot: BMI vs SleepTime by Heart Disease')
plt.xlabel('BMI')
plt.ylabel('SleepTime')
plt.show()
```




- The count plot with hue='Stroke' shows the proportion of people with heart disease who also suffered a stroke.
- The scatter plot shows how BMI and sleep time vary among those with and without heart disease, helping identify potential clusters or trends.

### *5. Encoding Categorical Data:*

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()

categorical_columns = df.select_dtypes(include=['object']).columns
df[categorical_columns] = df[categorical_columns].apply(lambda col: label_encoder.fit_transform(col))

df.head()
```

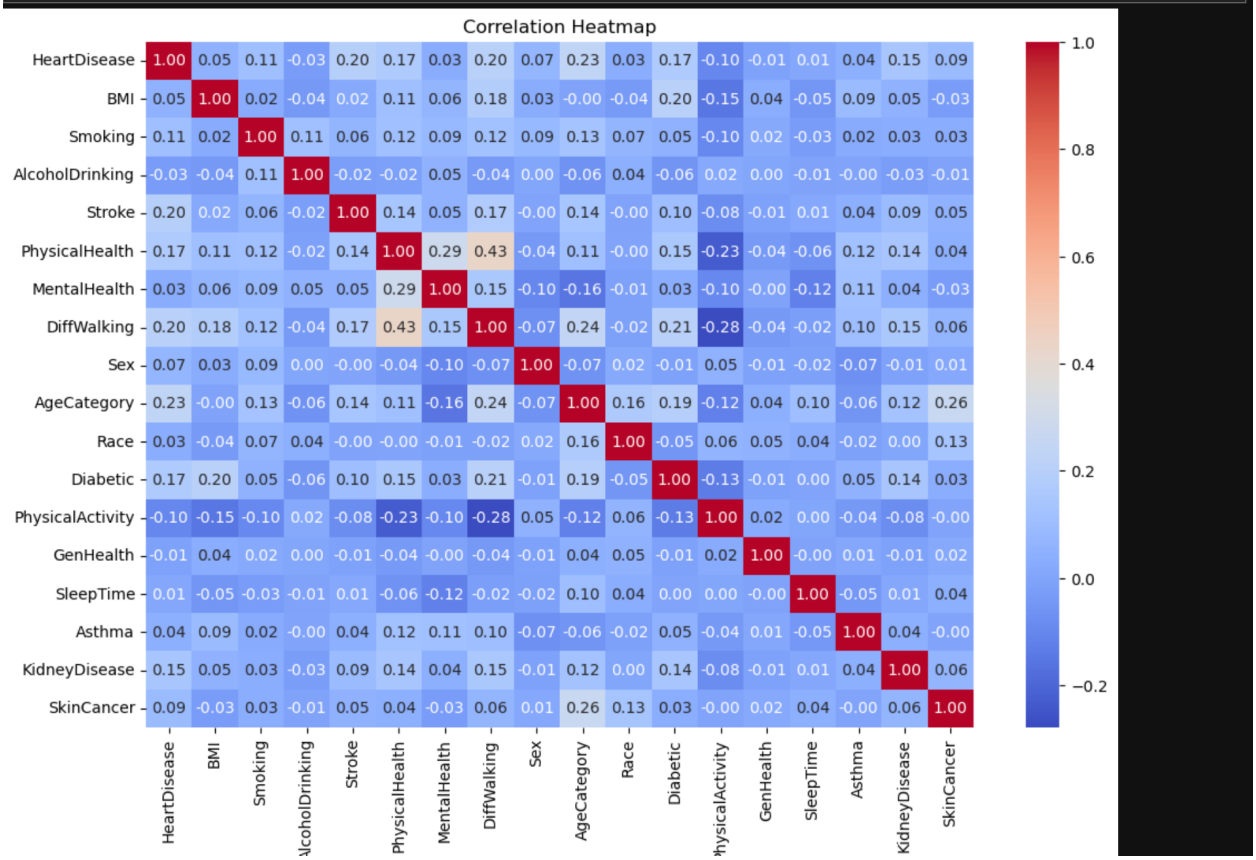
| | HeartDisease | BMI | Smoking | AlcoholDrinking | Stroke | PhysicalHealth | MentalHealth | DiffWalking | Sex | AgeCategory | Race | Diabetic | PhysicalActivity | GenH |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 16.60 | 1 | 0 | 0 | 3.0 | 30.0 | 0 | 0 | 7 | 5 | 2 | 1 | |
| **1** | 0 | 20.34 | 0 | 0 | 1 | 0.0 | 0.0 | 0 | 0 | 12 | 5 | 0 | 1 | |
| **2** | 0 | 26.58 | 1 | 0 | 0 | 20.0 | 30.0 | 0 | 1 | 9 | 5 | 2 | 1 | |
| **3** | 0 | 24.21 | 0 | 0 | 0 | 0.0 | 0.0 | 0 | 0 | 11 | 5 | 0 | 0 | |
| **4** | 0 | 23.71 | 0 | 0 | 0 | 28.0 | 0.0 | 1 | 0 | 4 | 5 | 0 | 1 | |

- Machine learning models require numerical input. Therefore, categorical variables (e.g., gender, race, yes/no responses) are converted into numeric codes using LabelEncoder.
- Each unique category is assigned a numerical value. For example, 'Male' might be 0 and 'Female' 1.
- This is applied across all object-type columns in the dataset.

### *6. Correlation Heatmap:*

```
correlation_matrix = df.corr()

plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f", cmap="coolwarm", cbar=True)
plt.title("Correlation Heatmap")
plt.show()
```




- A correlation matrix calculates how closely related two variables are. Values close to 1 or -1 imply strong relationships.
- This heatmap provides a visual overview of correlations among all numerical features.
- Features that are highly correlated with the target (HeartDisease) are considered more impactful.

## 7. Feature Selection and Scaling:

```
X = df.iloc[:,[2,4,5,7,9,11,12,16]]
y=df.iloc[:,0]
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
```

- Specific features are selected based on domain knowledge and correlation strength.
- StandardScaler is used to bring all features to the same scale, which is essential for gradient-based models like logistic regression to perform efficiently.
- Without scaling, features with larger magnitudes could dominate the learning process.

## 8. Train-Test Split and Model Training:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)
```

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()

model.fit(X_tr
```
Toggle output scrolling

- The dataset is split into training (75%) and testing (25%) sets using train_test_split.
- A logistic regression model is instantiated and trained using the .fit() function on the training data.
- Logistic regression is used here because it's efficient for binary classification problems like predicting the presence or absence of a disease.

## 9. Prediction and Evaluation:

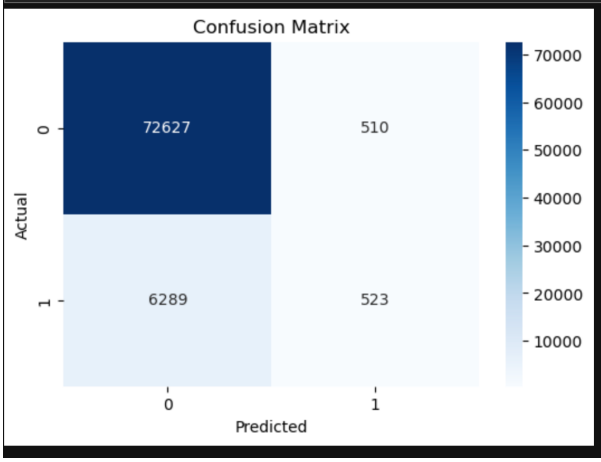
```
y_pred1=model.predict(X)
```

```
df['Prediction']=y_pred1
```

- Predictions are generated for the test data to evaluate how well the model generalizes.
- Predictions for the entire dataset are also stored in a new column Prediction for easy comparison.

## 12. Evaluation: Confusion Matrix and Classification Report:

```
# Confusion Matrix
from sklearn.metrics import confusion_matrix
plt.figure(figsize=(6, 4))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```




```
from sklearn.metrics import accuracy_score, classification_report
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.9149582859072659
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.99      0.96     73137
           1       0.51      0.08      0.13      6812

    accuracy                           0.91     79949
   macro avg       0.71      0.53      0.54     79949
weighted avg       0.89      0.91      0.89     79949
```

- The confusion matrix shows how many predictions were true positives, true negatives, false positives, and false negatives.
- Accuracy tells us the percentage of correct predictions.
- The classification report gives a breakdown of precision, recall, and F1-score for each class, helping evaluate model bias and performance on minority classes.

## Visualization and Model Evaluation:

### Data Visualizations

To gain insights into the dataset and understand the distribution of key variables, several visualizations were performed:

1. Count Plots:
    - The count plots of Stroke and HeartDisease show class distributions. Most individuals do not have a stroke or heart disease, indicating a class imbalance.
    - A bivariate count plot comparing heart disease cases with stroke history reveals a higher proportion of heart disease among stroke patients.
2. Pie Chart:
    - A pie chart of GenHealth indicates that the majority of individuals rate their health as either 'Good' or 'Very Good', while fewer report 'Poor' health.
3. Scatter Plot:
    - The scatter plot of BMI vs SleepTime colored by HeartDisease shows that most data points cluster around a BMI of 25–35 and sleep durations of 6–8 hours. However, no strong visual separation between those with and without heart disease is immediately evident.
4. Heatmap (Correlation Matrix):
    - A heatmap of feature correlations shows that PhysicalHealth, Stroke, and GenHealth have moderate correlations with HeartDisease. This helps prioritize features for modeling.

### Model Evaluation

Once the logistic regression model was trained on the selected features, it was evaluated using a set of standard classification metrics:

- Accuracy: The overall accuracy of the model was 91.49%, indicating that the model correctly predicts the outcome in a high percentage of cases.
- Confusion Matrix:
    - True Positives (TP): Correctly identified heart disease cases.
    - True Negatives (TN): Correctly identified non-heart disease cases.
    - False Positives (FP): Incorrectly predicted heart disease.
    - False Negatives (FN): Missed actual heart disease cases.
    - The confusion matrix showed a high number of true negatives but a low number of true positives, highlighting the challenge in detecting the minority class.
- Classification Report:
    - Precision (0.92) for non-heart disease cases was high, meaning predictions are usually correct when the model says "No Heart Disease".
    - Recall (0.99) for the majority class was excellent, but recall for heart disease (1) was only 0.08, meaning most actual positive cases were missed.
    - F1-Score for the heart disease class was low (0.13), confirming that the model struggles with imbalance.

This evaluation reveals that the logistic regression model performs well overall but is biased toward the majority class. This is a common issue in healthcare datasets, where disease cases are relatively rare.

## Conclusion and Insights

The project successfully demonstrates the process of predicting heart disease using logistic regression. From data cleaning and visualization to training and evaluation, each step highlights key practices in data science.

Key Takeaways:

- Logistic regression provides a strong baseline model with high overall accuracy.
- Visualizations helped uncover patterns, such as the link between stroke history and heart disease, and the prevalence of good self-reported health.
- Data imbalance is a major challenge. Although the model has high accuracy, it fails to detect most of the true positive heart disease cases.

Recommendations for Improvement:

- Class Imbalance: Use resampling techniques such as SMOTE or adjust class weights to improve recall for minority class.
- Model Alternatives: Explore more powerful algorithms like Random Forest, XGBoost, or Neural Networks which may capture complex patterns better.
- Feature Engineering: Combine or transform features for better separation (e.g., BMI categories, sleep quality index).
- Regularization and Hyperparameter Tuning: Fine-tune model parameters to improve generalization.

In conclusion, this project highlights the importance of data quality, balanced modeling, and critical evaluation. With further improvements, such models could support early detection and treatment planning for heart disease patients.

Github link - https://github.com/DakshrajKurki/IDS-assignment---2