HEART STROKE PREDICTION

GROUP- 16

PRESENTED BY: TEAM-16

MOHANA SAI KUMAR REDDY BOBBA

BINDU YADDULA

RADHA GUDE

ALEKHYA JILLELA

SINDHU VARDHAN MADALA

CONTENTS

- Abstract
- Introduction
- Problem Statement
- Problem description
- Environment Configuration
- Dataset
- Existing System and Proposed System
- Methodology
- Phases
- Installation of Pyspark and Libraries
- Creating Spark Session
- Result
- Plot count value Vs Heart Attack
- Plot count value Vs Education
- Plot count value Vs Income
- Conclusion
- References

ABSTRACT

- This project helps us to study stroke prediction. Stroke is the second leading disease that causes death in the world, according to the World Health Organization (WHO). Cardiovascular diseases (CVDs) kill about 20.5 million people every year. The project consists of many phases: cleaning the data sets, analysis, and analytics. We use big data tools for analyzing the data. The report provides the data on stroke that contains a person's information like gender, age, heart disease. We achieve the task of real-time analysis and implementing data with the help of bigdata infrastructure. our model showed us accurate results in terms of predicting the heart stroke of a patient with the parameters.

Keywords: Heart, Data, Classification, Machine learning, spark, Pyspark

INTRODUCTION

According to WHO, Heart Diseases are a leading cause of death worldwide. However, it is quite challenging to identify the cardiovascular disease (CVD) because of some contributory factors contributing to CVD like high blood pressure, cholesterol level, diabetics, abnormal pulse rate, and many other factors. In addition, sometimes CVD symptoms may vary for different genders. For example, a male patient is more likely to have chest pain while a female patient has some other symptoms with chest pain like chest discomfort: such as nausea, extreme fatigue, and shortness of breath.

PROBLEM STATEMENT

- The main challenge in today's healthcare is to provide high-quality services and accurate diagnoses. Even though heart disease has been identified as the leading cause of death worldwide in recent years, it is also one of the diseases that can be effectively controlled and managed. The accuracy of disease management depends on the proper time of disease detection. The proposed work aims to detect these heart diseases at an early stage to avoid disastrous consequences. The most challenging aspect of heart disease is its detection
- There are instruments that can predict heart disease, but they are either too expensive or too inefficient to calculate the risk of heart disease in humans. Early detection of cardiac diseases has been shown to reduce mortality and overall complications.
- However, it is impossible to monitor patients accurately every day, and consultation with a doctor for 24 hours is not available because it requires more intelligence, time, and expertise.

PROBLEM DESCRIPTION

- To implement a model on classifying a person can get a stroke based on his health and individual parameters
- Tools like spark and python has been used in order achieve the concept
- The idea is to classify and corelate the terms effecting the stroke, parameters which can have the impact.

ENVIRONMENT CONFIGURATION

- Google Colab, Python, Hadoop, PySpark and HDFS.
- Management of a Spark Session.
- Domain: Machine Learning / Deep Learning.
- Load different types of files (CSV and JSON).
- SQL queries with SparkSQL.
- Visualization with Matplotlib.

DATASET

- Kaggle dataset - https://www.kaggle.com/alexteboul/heart-disease-health-indicators-dataset
- This dataset contains 253,680 survey responses from cleaned BRFSS(Behavioral Risk Factor Surveillance System) 2015 to be used primarily for the binary classification of heart disease dataset. 229,787 respondents do not have/have not had heart disease while 23,893 have had heart disease.

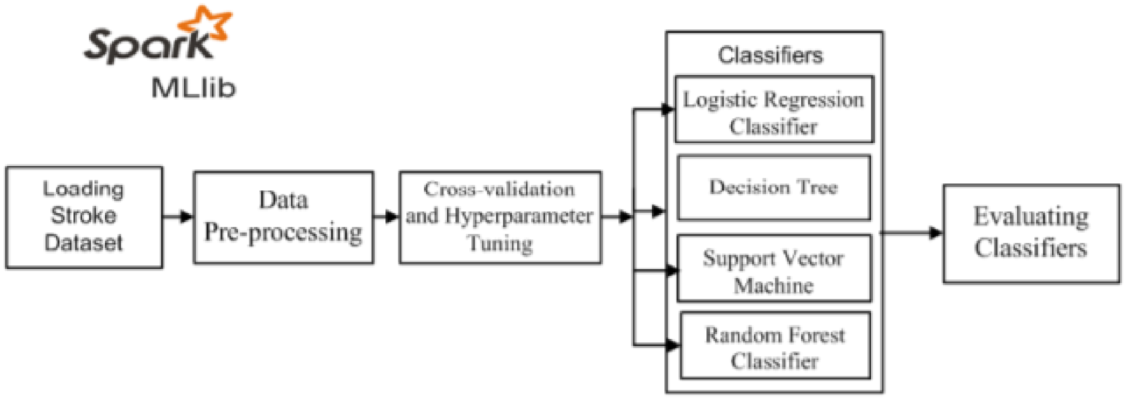
Cont'd

FEATURES	DESCRIPTION
Age	Age in number
Gender	Male or Female
Education rating	Scale of 1-5
Income	In number
BP	Blood Pressure
BMI	Body Mass Index
Heart Disease	Attacked or not

EXISTING SYSTEM

Due to ever-increasing medical data, we need to leverage machine learning algorithms to assist medical healthcare professionals in analyzing data and making accurate and precise diagnostic decisions. For example, in medical data mining, different classification algorithms are used to predict the CVD in patients and death predictions due to a heart attack.

PROPOSED SYSTEM



- The primary purpose is to highlight a comparison of several machine learning approaches to pick the best strategy for predicting heart disease survival. The first attempt is to apply machine learning models to all dataset features in predicting cardiac patients' survival. On the other hand, various optimization strategies have improved several metrics, including accuracy, precision, and recall.

METHODOLOGY

Step1: Collect and store the Data.

Step2: Data preprocessing.

Step3: Extract the features from the dataset.

Step4: Make a model to learn from the different input features.

Step5: Predict the results.

PHASES

- 1. Inserting the tools required in our system
- 2. Collect and clean the datasets
- 3. Concentration on tools required
- 4. Importing the data by Spark setup
- 5. Queries implementation with visualization in big data tools
- 6. Analyzing the data and
- 7. Data analytics

PHASE 1

Installation of necessary tools into our equipment.

Appropriate data sets must be gathered.

Working on the requirements for functionality.

PHASE 2

Importing the libraries that are required.

The data is loaded into the Spark server after it is created.

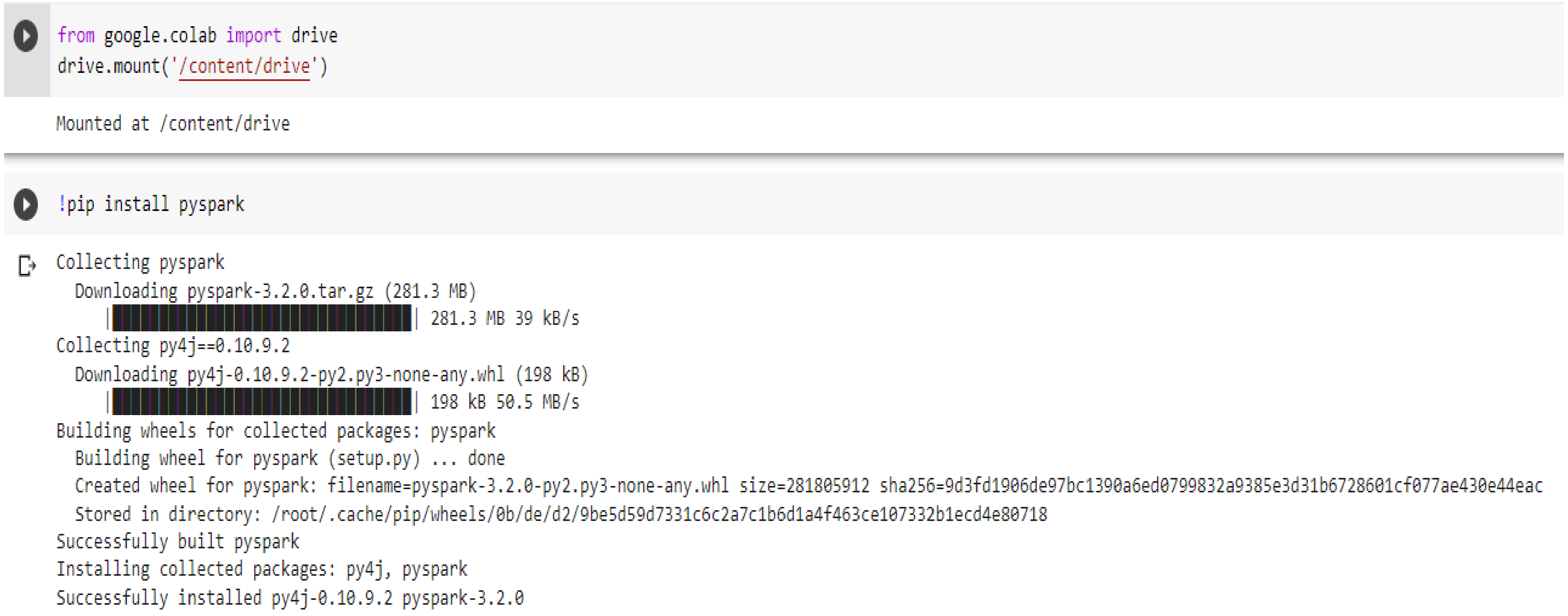
Data Cleaning

PHASE 3

About the data, data preprocessing

Visualization of data

INSTALLATION OF PYSPARK



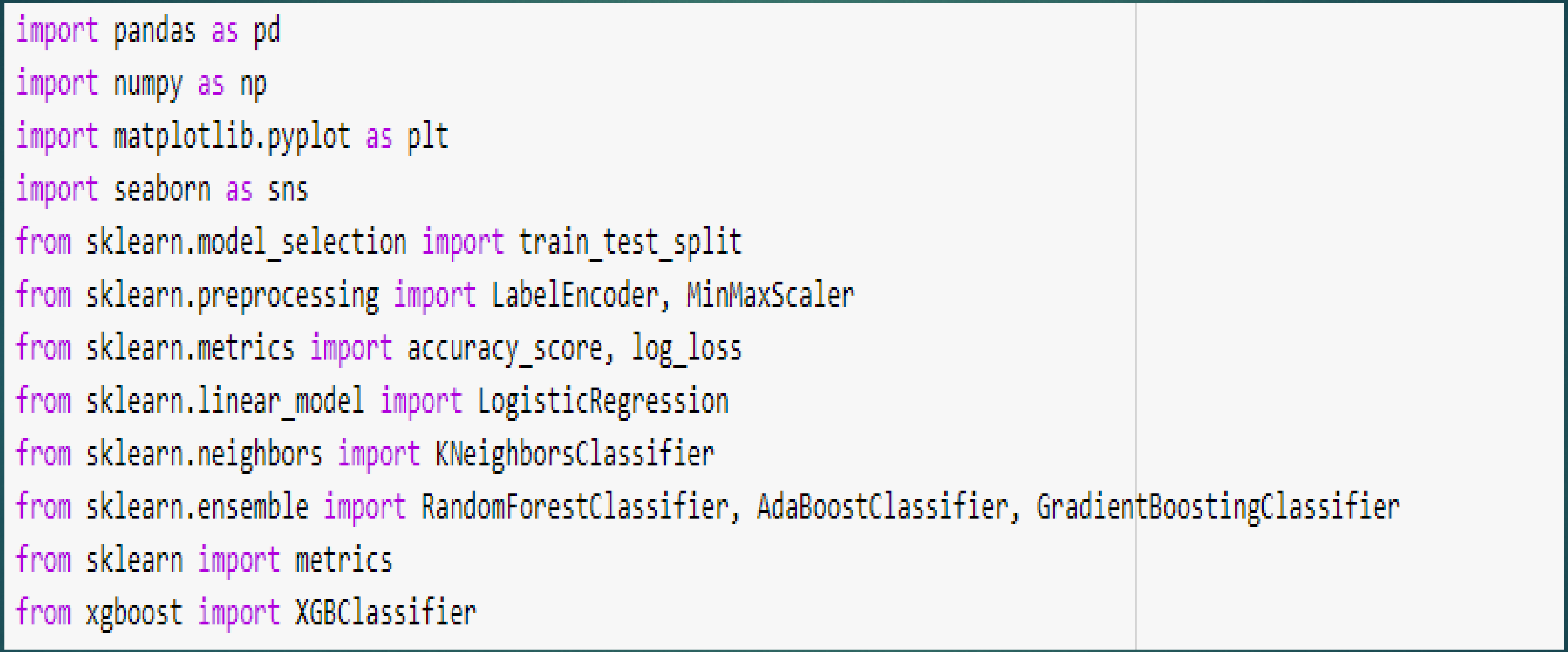
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
!pip install pyspark
```

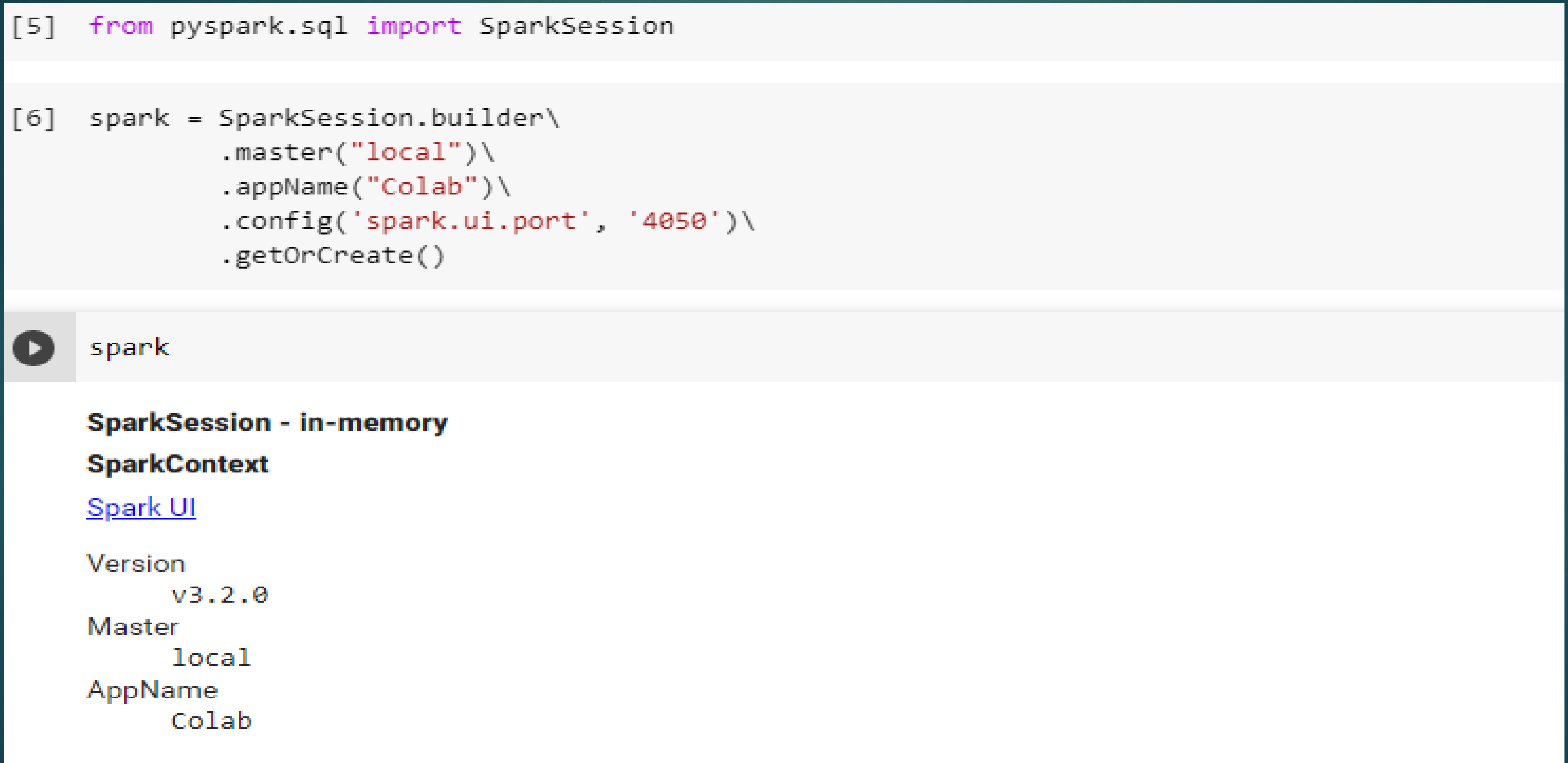
```
Collecting pyspark
  Downloading pyspark-3.2.0.tar.gz (281.3 MB)
     |████████████████████████████████| 281.3 MB 39 kB/s
Collecting py4j==0.10.9.2
  Downloading py4j-0.10.9.2-py2.py3-none-any.whl (198 kB)
     |████████████████████████████████| 198 kB 50.5 MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.2.0-py2.py3-none-any.whl size=281805912 sha256=9d3fd1906de97bc1390a6ed0799832a9385e3d31b6728601cf077ae430e44eac
  Stored in directory: /root/.cache/pip/wheels/0b/de/d2/9be5d59d7331c6c2a7c1b6d1a4f463ce107332b1ecd4e80718
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.2 pyspark-3.2.0
```

LIBRARIES



```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.metrics import accuracy_score, log_loss
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier, GradientBoostingClassifier
from sklearn import metrics
from xgboost import XGBClassifier
```

CREATING SPARK SESSION



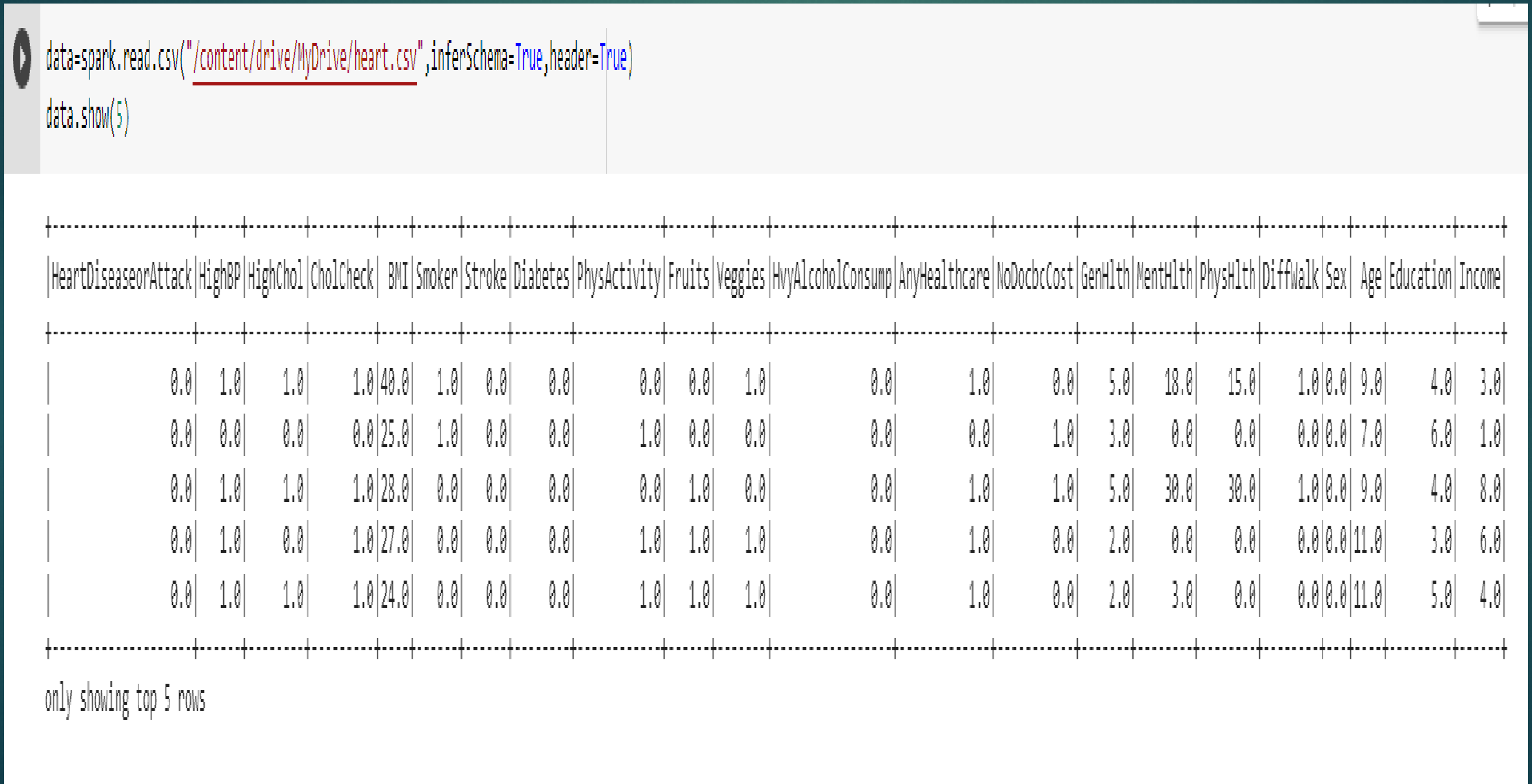
```
[5]  from pyspark.sql import SparkSession
```

```
[6]  spark = SparkSession.builder\
             .master("local")\
             .appName("Colab")\
             .config('spark.ui.port', '4050')\
             .getOrCreate()
```

```
spark
```

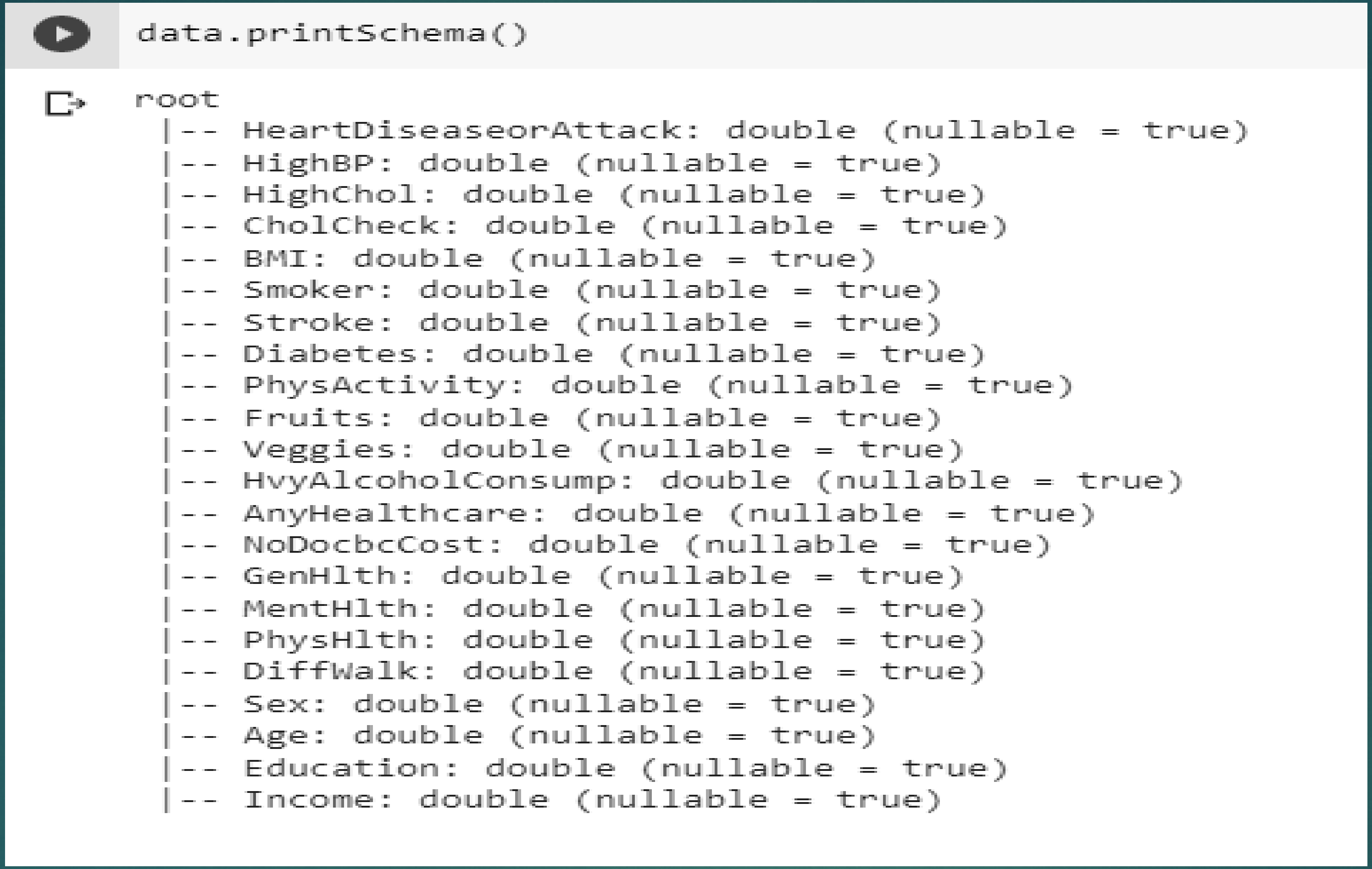
```
SparkSession - in-memory
SparkContext
Spark UI

Version
        v3.2.0
Master
        local
AppName
        Colab
```

RESULT



```
data=spark.read.csv("/content/drive/MyDrive/heart.csv",inferSchema=True,header=True)
data.show(5)
```

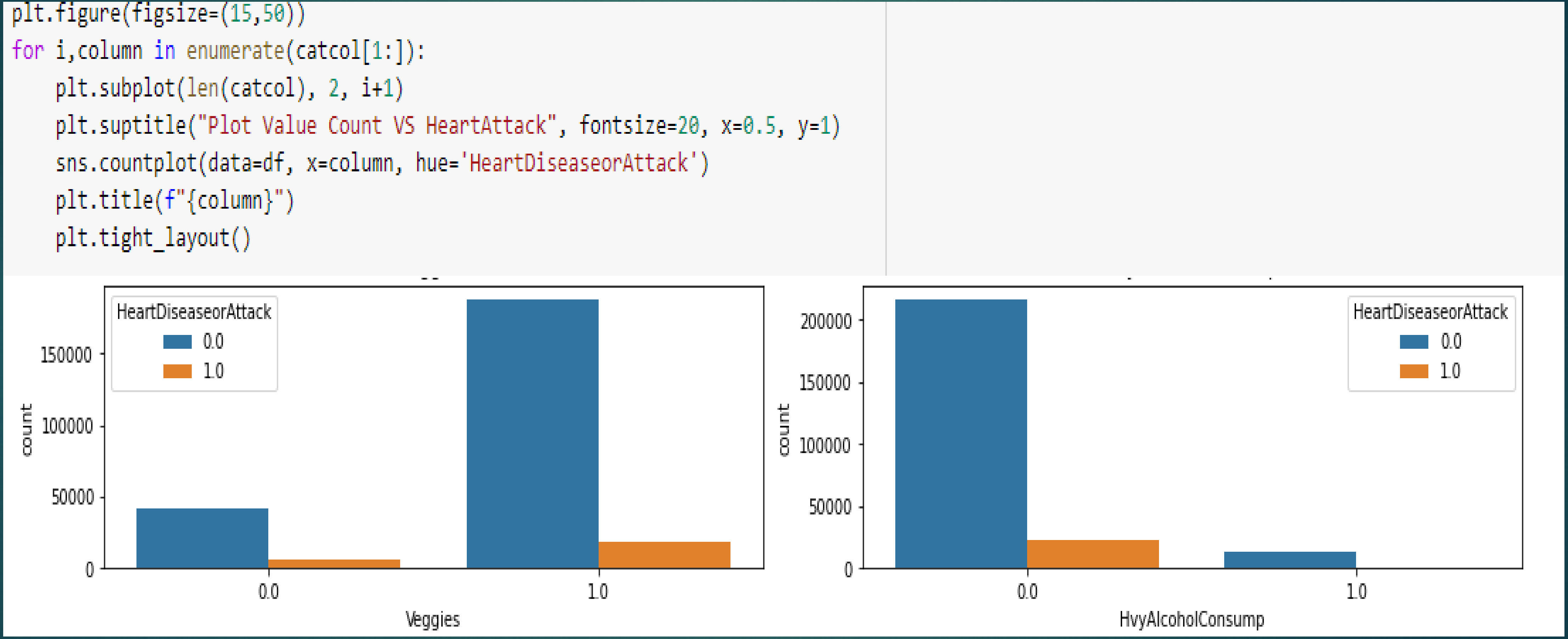
```
+--------------------+------+--------+---------+----+------+------+--------+------------+------+-------+-----------------+-------------+-----------+-------+--------+--------+--------+---+----+---------+------+
|HeartDiseaseorAttack|HighBP|HighChol|CholCheck| BMI|Smoker|Stroke|Diabetes|PhysActivity|Fruits|Veggies|HvyAlcoholConsump|AnyHealthcare|NoDocbcCost|GenHlth|MentHlth|PhysHlth|DiffWalk|Sex| Age|Education|Income|
+--------------------+------+--------+---------+----+------+------+--------+------------+------+-------+-----------------+-------------+-----------+-------+--------+--------+--------+---+----+---------+------+
|                 0.0|   1.0|     1.0|      1.0|40.0|   1.0|   0.0|     0.0|         0.0|   0.0|    1.0|              0.0|          1.0|        0.0|    5.0|    18.0|    15.0|     1.0|0.0| 9.0|      4.0|   3.0|
|                 0.0|   0.0|     0.0|      0.0|25.0|   1.0|   0.0|     0.0|         1.0|   0.0|    0.0|              0.0|          0.0|        1.0|    3.0|     0.0|     0.0|     0.0|0.0| 7.0|      6.0|   1.0|
|                 0.0|   1.0|     1.0|      1.0|28.0|   0.0|   0.0|     0.0|         0.0|   1.0|    0.0|              0.0|          1.0|        1.0|    5.0|    30.0|    30.0|     1.0|0.0| 9.0|      4.0|   8.0|
|                 0.0|   1.0|     0.0|      1.0|27.0|   0.0|   0.0|     0.0|         1.0|   1.0|    1.0|              0.0|          1.0|        0.0|    2.0|     0.0|     0.0|     0.0|0.0|11.0|      3.0|   6.0|
|                 0.0|   1.0|     1.0|      1.0|24.0|   0.0|   0.0|     0.0|         1.0|   1.0|    1.0|              0.0|          1.0|        0.0|    2.0|     3.0|     0.0|     0.0|0.0|11.0|      5.0|   4.0|
+--------------------+------+--------+---------+----+------+------+--------+------------+------+-------+-----------------+-------------+-----------+-------+--------+--------+--------+---+----+---------+------+
only showing top 5 rows
```



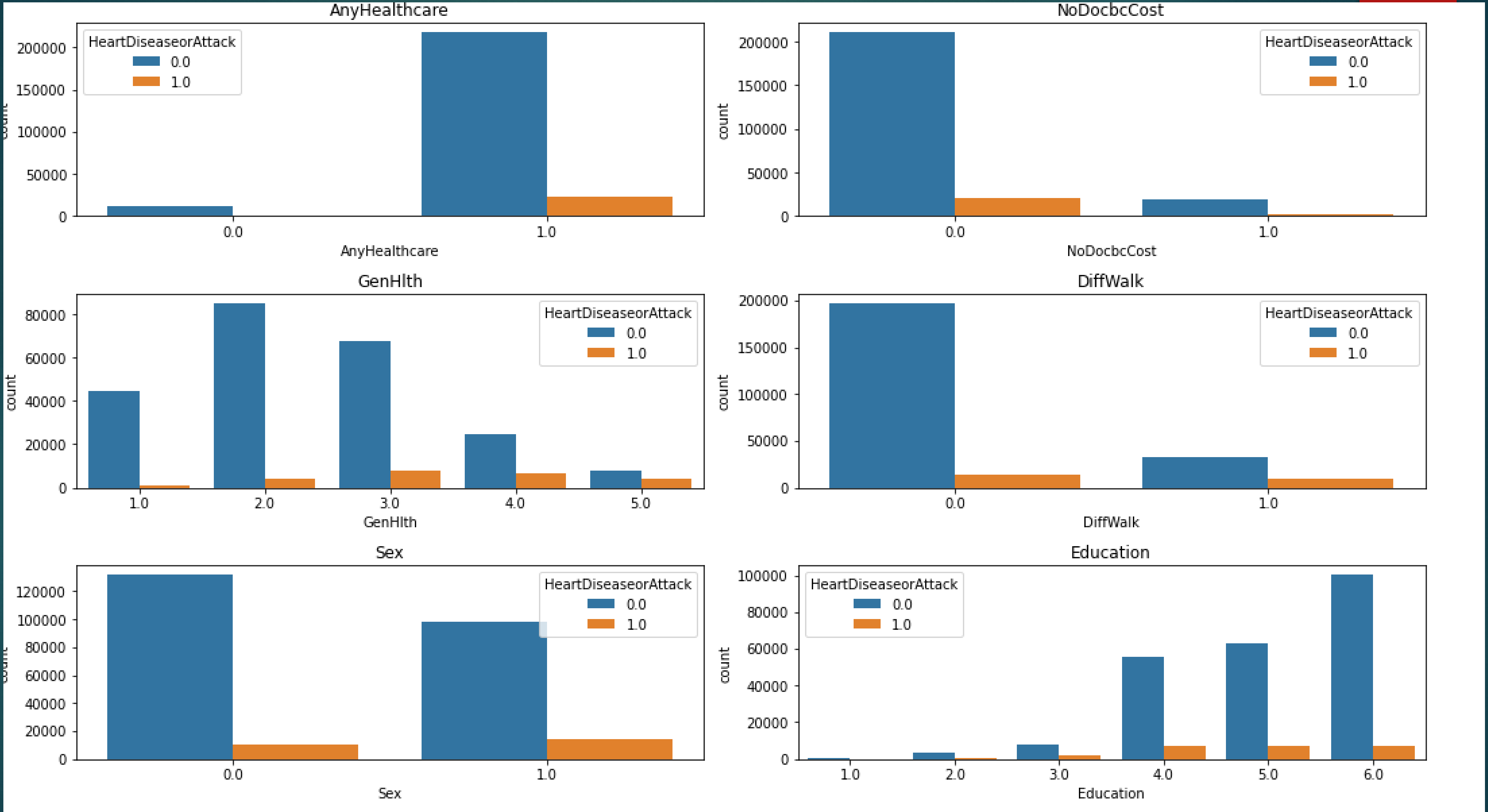
```
data.printSchema()
```

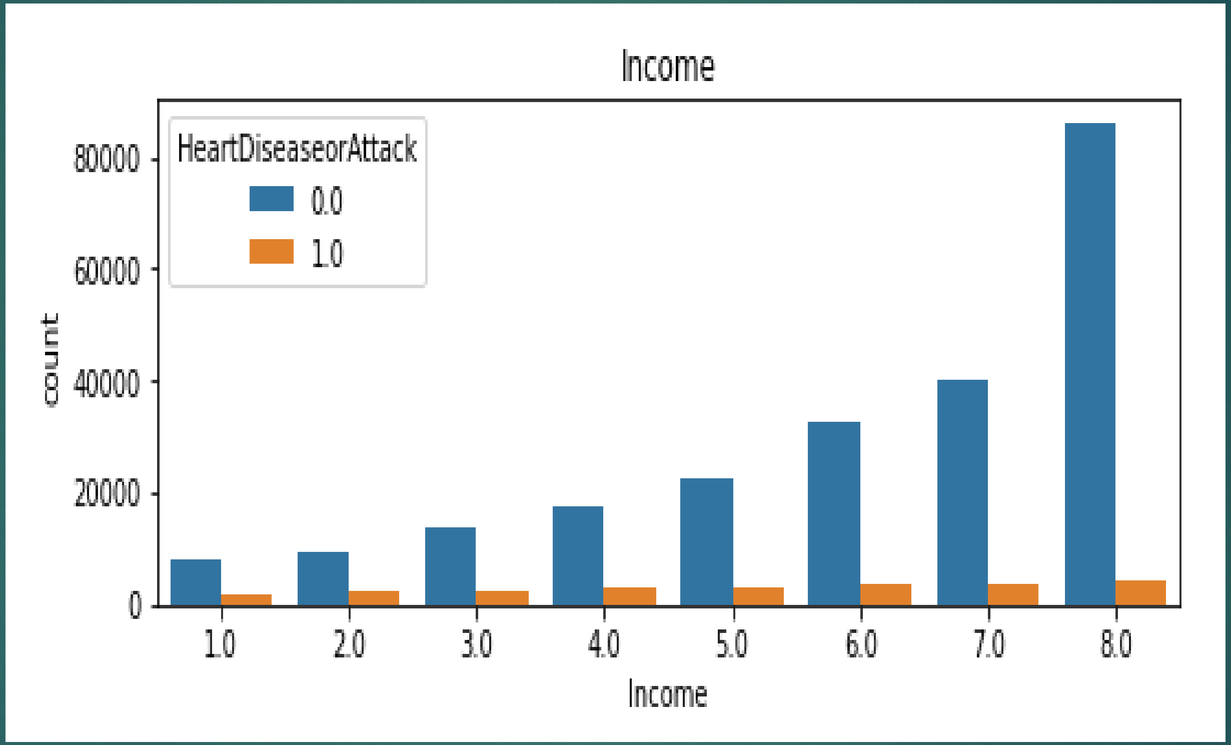
```
root
 |-- HeartDiseaseorAttack: double (nullable = true)
 |-- HighBP: double (nullable = true)
 |-- HighChol: double (nullable = true)
 |-- CholCheck: double (nullable = true)
 |-- BMI: double (nullable = true)
 |-- Smoker: double (nullable = true)
 |-- Stroke: double (nullable = true)
 |-- Diabetes: double (nullable = true)
 |-- PhysActivity: double (nullable = true)
 |-- Fruits: double (nullable = true)
 |-- Veggies: double (nullable = true)
 |-- HvyAlcoholConsump: double (nullable = true)
 |-- AnyHealthcare: double (nullable = true)
 |-- NoDocbcCost: double (nullable = true)
 |-- GenHlth: double (nullable = true)
 |-- MentHlth: double (nullable = true)
 |-- PhysHlth: double (nullable = true)
 |-- DiffWalk: double (nullable = true)
 |-- Sex: double (nullable = true)
 |-- Age: double (nullable = true)
 |-- Education: double (nullable = true)
 |-- Income: double (nullable = true)
```

PLOT COUNT VALUE VS HEART ATTACK

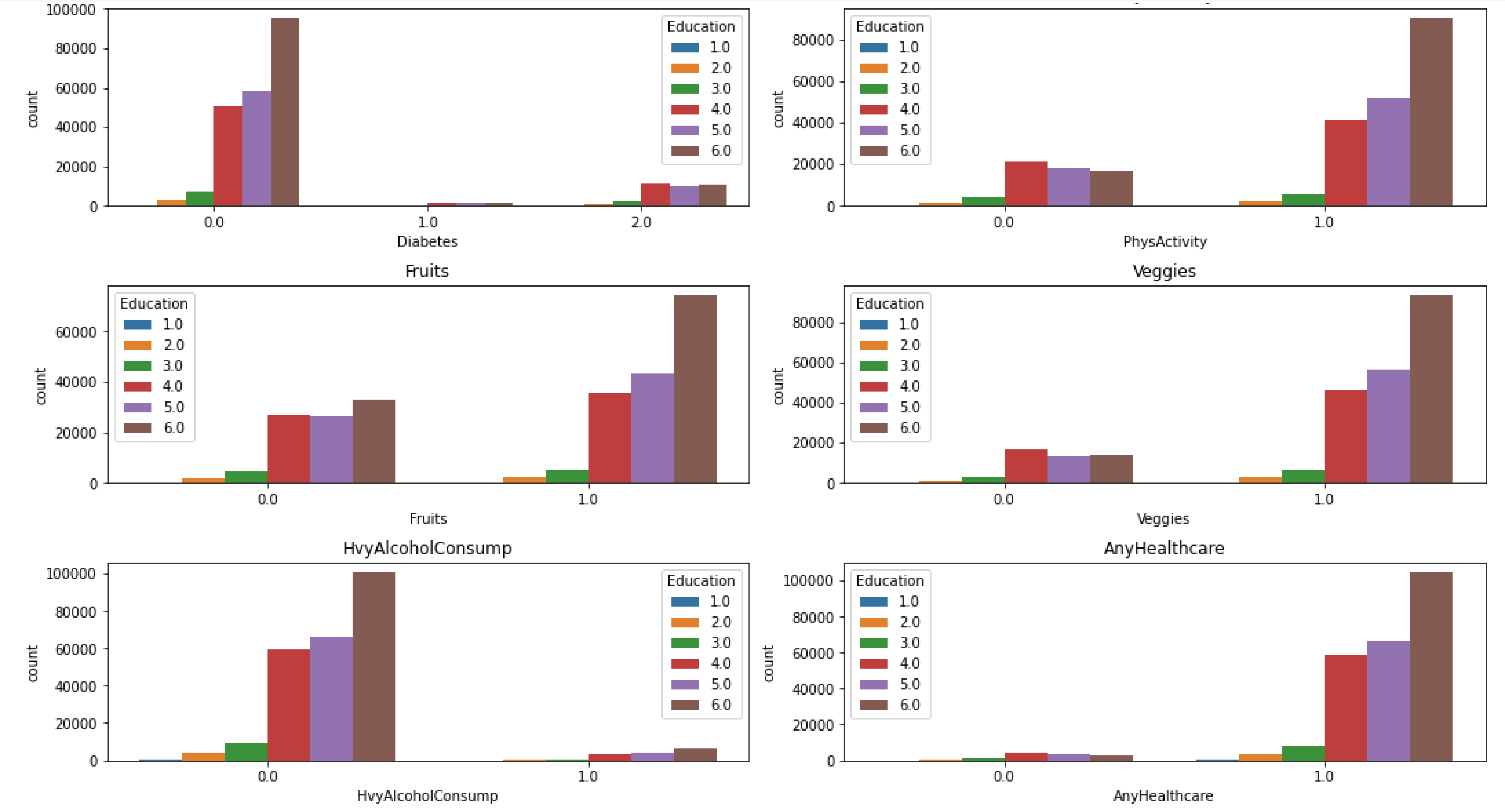


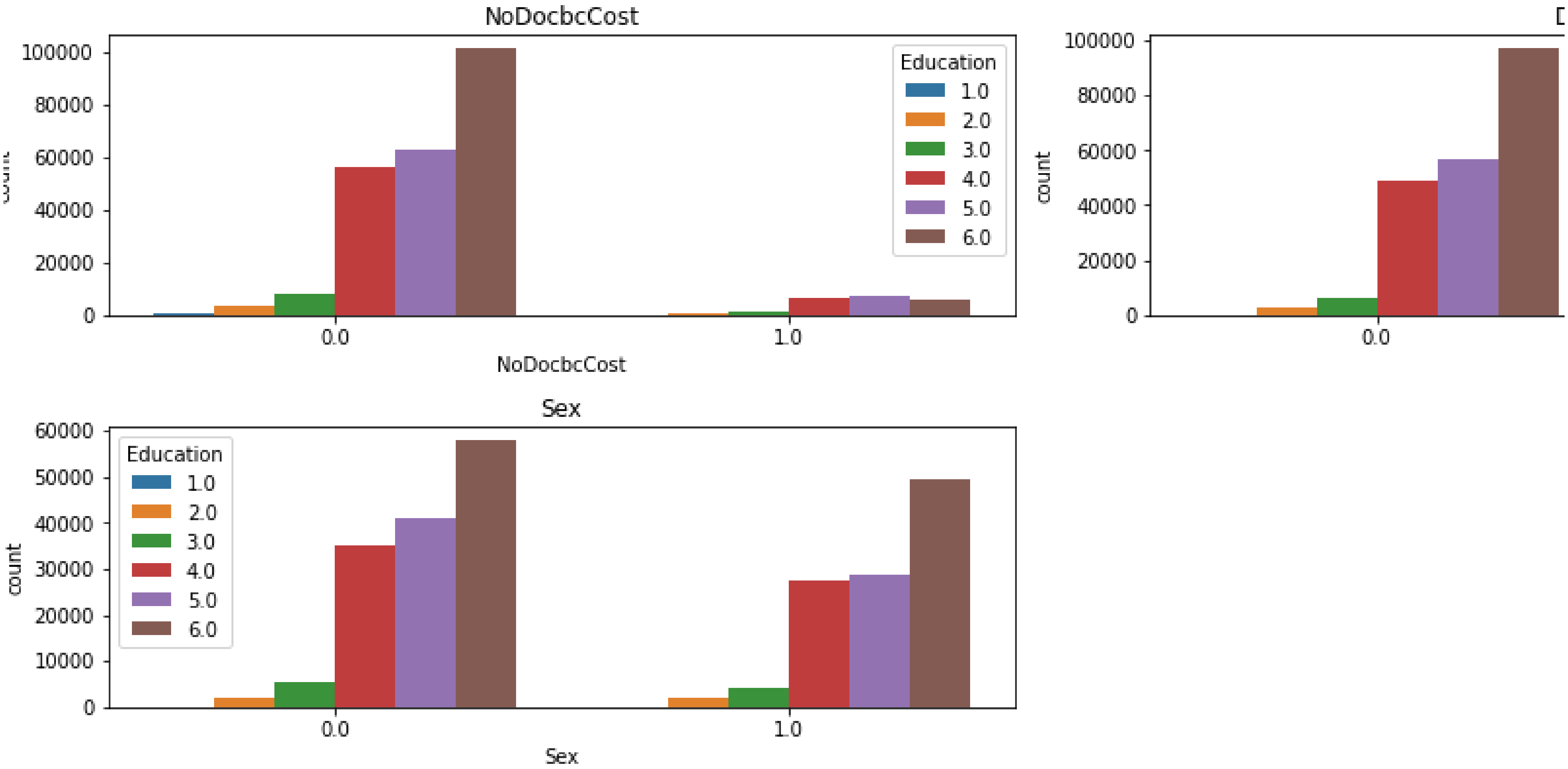
```
plt.figure(figsize=(15,50))
for i,column in enumerate(catcol[1:]):
    plt.subplot(len(catcol), 2, i+1)
    plt.suptitle("Plot Value Count VS HeartAttack", fontsize=20, x=0.5, y=1)
    sns.countplot(data=df, x=column, hue='HeartDiseaseorAttack')
    plt.title(f"{column}")
    plt.tight_layout()
```


AnyHealthcare
HeartDiseaseorAttack
🟦 0.0
🟧 1.0
count
0
50000
100000
150000
200000
0.0
1.0
AnyHealthcare
NoDocbcCost
HeartDiseaseorAttack
🟦 0.0
🟧 1.0
count
0
50000
100000
150000
200000
0.0
1.0
NoDocbcCost
GenHlth
HeartDiseaseorAttack
🟦 0.0
🟧 1.0
count
0
20000
40000
60000
80000
1.0
2.0
3.0
4.0
5.0
GenHlth
DiffWalk
HeartDiseaseorAttack
🟦 0.0
🟧 1.0
count
0
50000
100000
150000
200000
0.0
1.0
DiffWalk
Sex
HeartDiseaseorAttack
🟦 0.0
🟧 1.0
count
0
20000
40000
60000
80000
100000
120000
0.0
1.0
Sex
Education
HeartDiseaseorAttack
🟦 0.0
🟧 1.0
count
0
20000
40000
60000
80000
100000
1.0
2.0
3.0
4.0
5.0
6.0
Education

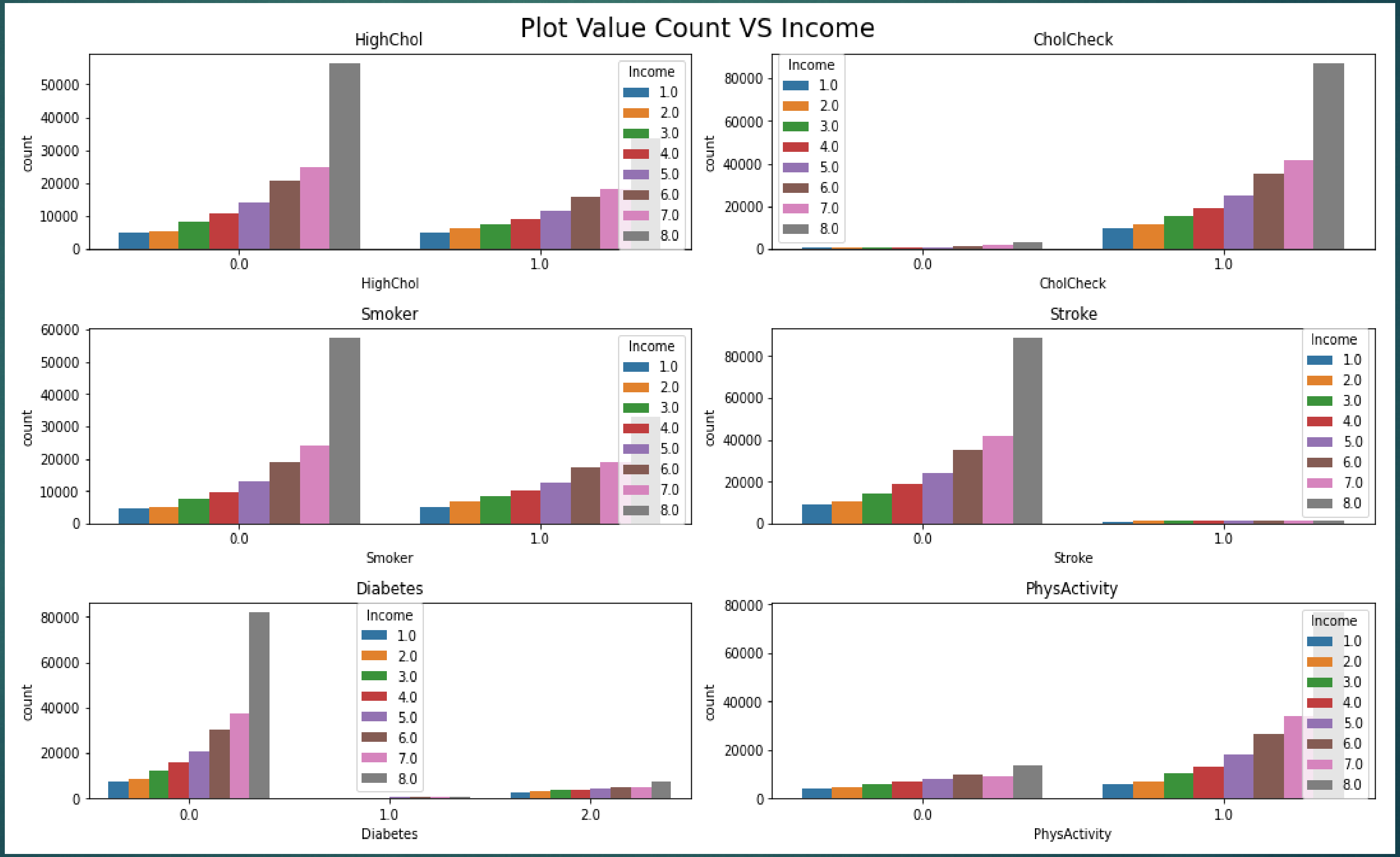

Income
HeartDiseaseorAttack
0.0
1.0
count
80000
60000
40000
20000
0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
Income

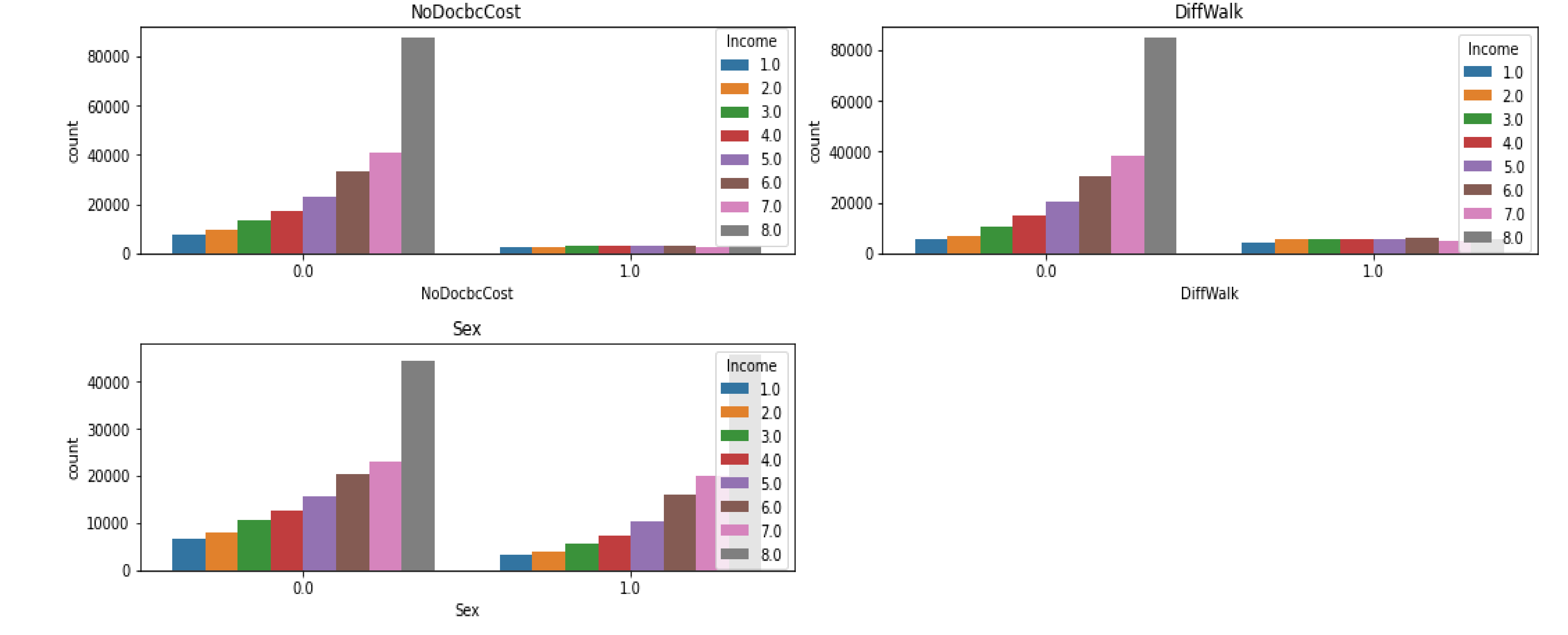
PLOT COUNT VALUE VS EDUCATION




NoDocbcCost
100000
80000
60000
40000
20000
0
count
Education
1.0
2.0
3.0
4.0
5.0
6.0
0.0
1.0
NoDocbcCost
D
100000
80000
60000
40000
20000
0
count
0.0
Sex
60000
50000
40000
30000
20000
10000
0
count
Education
1.0
2.0
3.0
4.0
5.0
6.0
0.0
1.0
Sex

PLOT VALUE VS HEART ATTACK BY INCOME




NoDocbcCost
count
80000
60000
40000
20000
0
0.0
1.0
NoDocbcCost
Income
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
DiffWalk
count
80000
60000
40000
20000
0
0.0
1.0
DiffWalk
Income
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
Sex
count
40000
30000
20000
10000
0
0.0
1.0
Sex
Income
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0

CONCLUSION

- The work begins with analyzing the required libraries for the model to build; later, we started to look after the dataset, with around 22 features and almost six lakhs' lines of data.
- Spark integrated with python has been installed in the working environment of google collab, where our data has been loaded from google drive. The model runs on different algorithms such as random forest, logistic regression.
- To analyze the data set, the data has been divided into training and testing parts, the training data is 80 percent, and the testing data is 20 percent. Our model showed us accurate results in predicting the heart stroke of a patient with the available parameters. The accuracy is well achieved with the random forest model.

REFERENCES

[1] L. Amini, R. Azarpazhouh, M. T. Farzadfar, S. A. Mousavi, F. Jazaieri, F. Khorvash, R. Norouzi, and N. Toghianfar, “Prediction and control of heart stroke by data mining,” International Journal of Preventive Medicine, vol. 4, no. Suppl 2, pp. S245–249, May 2013.

[2] S.-F. Sung, C.-Y. Hsieh, Y.-H. Kao Yang, H.-J. Lin, C.-H. Chen, Y.- W. Chen, and Y.-H. Hu, “Developing a heart stroke Severity index based on administrative data was feasible using data mining techniques,” Journal of Clinical Epidemiology, vol. 68, no. 11, pp. 1292–1300, Nov. 2015.

[5] P. Govindarajan, R. K. Soundarapandian, A. H. Gandomi, R. Patan, P. Jayaraman, and R. Manikandan, “Classification of heart stroke disease using machine learning algorithms,” Neural Computing and Applications, vol. 32, no. 3, pp. 817–828, Feb. 2020.

[6] C.V. Krishna Veni, T. R. Shoba, On the classification of imbalanced Datasets, International Journal of Computer Science & Technology 2011; 2:145-148

THANK YOU