Heart Disease Prediction System

A Comprehensive Report by Navneet Rahul

Batch-SB-23-9-5 MLI

Great Lakes of Institute of Management- Bangalore

Abstract:

The dataset provided for the heart disease prediction project is a valuable resource for building a robust predictive model. It contains a diverse set of attributes related to individuals' health and lifestyle, making it conducive for comprehensive analysis. Here's an extract highlighting key aspects of the dataset:

The dataset comprises several columns, each representing a specific attribute or characteristic related to the individuals under study. These attributes include:

1. Heart Disease: This binary classification target variable indicates whether an individual has been diagnosed with heart disease or not. It serves as the primary outcome variable for our prediction model.

2. BMI: Body Mass Index (BMI) is a measure of an individual's body fat based on height and weight. It provides insights into an individual's overall health and can be a significant predictor of heart disease risk.

3. Smoking: This binary variable indicates whether an individual is a smoker or not. Smoking is a well-established risk factor for heart disease.

4. Alcohol Drinking: Similar to smoking, alcohol consumption can impact heart health. This variable denotes whether an individual drinks alcohol regularly.

5. Stroke, Physical Health, Mental Health, DiffWalking: These attributes provide information about an individual's overall health and well-being, which can be associated with heart disease risk.

6. Sex and Age Category: Demographic variables like gender and age are crucial factors in heart disease risk assessment. Age, in particular, is a significant risk factor.

7. Race: The race or ethnicity of the individuals can be relevant as heart disease prevalence can vary among different racial groups.

8. Diabetic, Physical Activity, GenHealth, Sleep Time, Asthma, Kidney Disease, Skin Cancer: These variables encompass various aspects of an individual's medical history, lifestyle, and health conditions, all of which can contribute to the prediction of heart disease risk.

The dataset is diverse, containing both categorical and continuous variables, allowing for a multifaceted analysis. By exploring and preprocessing this dataset, we can develop a predictive model that takes into account a wide range of factors to accurately classify individuals' heart disease risk. This dataset's richness makes it a valuable asset for our project, enabling us to contribute to early heart disease detection and prevention.

Title: Heart Disease Prediction System - A Comprehensive Report

1. Problem Statement:

The project aims to develop a heart disease prediction system using a dataset that includes various health-related attributes such as BMI, smoking habits, alcohol consumption, and more. The goal is to build a predictive model that can accurately classify individuals as either having heart disease or not. This system can assist healthcare professionals in early diagnosis and intervention.

2. Market/Customer/Business Need Assessment:

The need for a reliable heart disease prediction system is evident due to the rising prevalence of heart-related illnesses. Cardiovascular diseases are a leading cause of death worldwide. Early detection and intervention can significantly reduce mortality rates and healthcare costs. This project addresses the growing demand for accurate and accessible predictive tools in the healthcare sector.

3. Target Specifications and Characterization:

The target audience for this system includes healthcare providers, medical researchers, and individuals interested in monitoring their heart health. Healthcare providers can use it to identify high-risk patients, while individuals can assess their own risk factors.

4. External Search:

A comprehensive review of existing literature, research papers, and online resources related to heart disease prediction, machine learning algorithms, and healthcare datasets was conducted. The research findings informed the development process.

The dataset can be found on the Kaggle.

```python
df=pd.read_csv('heart_2020_cleaned.csv')
```

```python
df.head()
```

	HeartDisease	BMI	Smoking	AlcoholDrinking	Stroke	PhysicalHealth	MentalHealth	DiffWalking	Sex	AgeCategory	Race	Diabetic	PhysicalActivity	Ge
0	No	16.60	Yes	No	No	3.0	30.0	No	Female	55-59	White	Yes	Yes	V
1	No	20.34	No	No	Yes	0.0	0.0	No	Female	80 or older	White	No	Yes	V
2	No	26.58	Yes	No	No	20.0	30.0	No	Male	65-69	White	Yes	Yes	
3	No	24.21	No	No	No	0.0	0.0	No	Female	75-79	White	No	No	
4	No	23.71	No	No	No	28.0	0.0	Yes	Female	40-44	White	No	Yes	V

Attributes of the data

1. HeartDisease- It is a binary variable, indicating whether the individual has heart disease ("Yes" or "No").
2. BMI- BMI represents Body Mass Index, a measure of obesity or overweight.
3. "Smoking" and "AlcoholDrinking" represent smoking and alcohol consumption habits.
4. "PhysicalHealth" and "MentalHealth" appear to be health-related ratings or scores.
5. "AgeCategory" categorizes individuals into age groups.
6. "Diabetic," "Asthma," "KidneyDisease," and "SkinCancer" indicate the presence or absence of these health conditions.
7. "Stroke" is also a binary variable, indicating whether the individual has had a stroke ("Yes" or "No").

Step2 : Data Sanity check

- Get the basic info of the data.
- Look for null values
- Look for corrupted data
- Get the data summary statistics (both numerical and categorical)
- Look for erroneous values in the data

```python
#Get the shape of the data
data_shape=df.shape
print("Rows= ",data_shape[0],"\nColumns =",data_shape[1])
```

```
Rows=  319795
Columns = 18
```

```python
#Get the basic info
info=df.info()

#get the data type
dtype=df.dtypes
info,dtype
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 319795 entries, 0 to 319794
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   HeartDisease      319795 non-null  object 
 1   BMI               319795 non-null  float64
 2   Smoking           319795 non-null  object 
 3   AlcoholDrinking   319795 non-null  object 
 4   Stroke            319795 non-null  object 
 5   PhysicalHealth    319795 non-null  float64
 6   MentalHealth      319795 non-null  float64
 7   DiffWalking       319795 non-null  object 
 8   Sex               319795 non-null  object 
 9   AgeCategory       319795 non-null  object 
 10  Race              319795 non-null  object 
 11  Diabetic          319795 non-null  object 
 12  PhysicalActivity  319795 non-null  object 
 13  GenHealth         319795 non-null  object 
 14  SleepTime         319795 non-null  float64
 15  Asthma            319795 non-null  object 
 16  KidneyDisease     319795 non-null  object 
 17  SkinCancer        319795 non-null  object 
dtypes: float64(4), object(14)
memory usage: 43.9+ MB
```

```
#Check for unique levels in categorical
df.Stroke.unique()
```

```
array(['No', 'Yes'], dtype=object)
```

```
#Check for nulls and duplicates
nulls=df.isnull().sum()
dups=df.duplicated().sum()
nulls,dups
```

```
(HeartDisease        0
 BMI                 0
 Smoking             0
 AlcoholDrinking     0
 Stroke              0
 PhysicalHealth      0
 MentalHealth        0
 DiffWalking         0
 Sex                 0
 AgeCategory         0
 Race                0
 Diabetic            0
 PhysicalActivity    0
 GenHealth           0
 SleepTime           0
 Asthma              0
 KidneyDisease       0
 SkinCancer          0
 dtype: int64,
 18078)
```

Dataset Link- https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease/data

```
#Summary statistics of numerical and categorical data
num_stats=df.describe().T
cat_stats=df.describe(include='O').T
print(num_stats)
print(cat_stats)
```

```
                  count       mean       std    min    25%    50%    75%  \
BMI            319795.0  28.325399  6.356100  12.02  24.03  27.34  31.42
PhysicalHealth 319795.0   3.371710  7.950850   0.00   0.00   0.00   2.00
MentalHealth   319795.0   3.898366  7.955235   0.00   0.00   0.00   3.00
SleepTime      319795.0   7.097075  1.436007   1.00   6.00   7.00   8.00

                 max
BMI            94.85
PhysicalHealth 30.00
MentalHealth   30.00
SleepTime      24.00
                  count unique        top    freq
HeartDisease     319795      2         No  292422
Smoking          319795      2         No  187887
AlcoholDrinking  319795      2         No  298018
Stroke           319795      2         No  307726
DiffWalking      319795      2         No  275385
Sex              319795      2     Female  167805
AgeCategory      319795     13      65-69   34151
Race             319795      6      White  245212
Diabetic         319795      4         No  269653
PhysicalActivity 319795      2        Yes  247957
GenHealth        319795      5  Very good  113858
Asthma           319795      2         No  276923
KidneyDisease    319795      2         No  308016
SkinCancer       319795      2         No  289976
```

Step 3: Data Cleaning Step

- AgeCategory shouldn't be categorical, so I will apply a function to calculate the mean age and make it a continuous feature

```
encode_AgeCategory = {'55-59':57, '80 or older':80, '65-69':67,
                      '75-79':77,'40-44':42,'70-74':72,'60-64':62,
                      '50-54':52,'45-49':47,'18-24':21,'35-39':37,
                      '30-34':32,'25-29':27}
df['AgeCategory'] = df['AgeCategory'].apply(lambda x: encode_AgeCategory[x])
df['AgeCategory'] = df['AgeCategory'].astype('float')
```

```
cat_col=df.select_dtypes(exclude=np.number)
num_col=df.select_dtypes(include=np.number)
print(cat_col.columns)
print(num_col.columns)
```

```
Index(['HeartDisease', 'Smoking', 'AlcoholDrinking', 'Stroke', 'DiffWalking',
       'Sex', 'Race', 'Diabetic', 'PhysicalActivity', 'GenHealth', 'Asthma',
       'KidneyDisease', 'SkinCancer'],
      dtype='object')
Index(['BMI', 'PhysicalHealth', 'MentalHealth', 'AgeCategory', 'SleepTime'], dtype='object')
```

```
df['HeartDisease'].value_counts()
```

```
No     292422
Yes     27373
Name: HeartDisease, dtype: int64
```

```python
#Target Variable
df['HeartDisease'].value_counts()
```

```
No     292422
Yes     27373
Name: HeartDisease, dtype: int64
```

```python
for col in cat_col:
    print("Value counts for column:", col)
    print(df[col].value_counts())
    print("\n")
```

```
Value counts for column: HeartDisease
No     292422
Yes     27373
Name: HeartDisease, dtype: int64


Value counts for column: Smoking
No     187887
Yes    131908
Name: Smoking, dtype: int64


Value counts for column: AlcoholDrinking
No     298018
Yes     21777
Name: AlcoholDrinking, dtype: int64
```

```python
#Creating a copy
data=df.copy()
data.head()
```

	HeartDisease	BMI	Smoking	AlcoholDrinking	Stroke	PhysicalHealth	MentalHealth	DiffWalking	Sex	AgeCategory	Race	Diabetic	PhysicalActivity	Ge
0	No	16.60	Yes	No	No	3.0	30.0	No	Female	57.0	White	Yes	Yes	V
1	No	20.34	No	No	Yes	0.0	0.0	No	Female	80.0	White	No	Yes	V
2	No	26.58	Yes	No	No	20.0	30.0	No	Male	67.0	White	Yes	Yes	
3	No	24.21	No	No	No	0.0	0.0	No	Female	77.0	White	No	No	
4	No	23.71	No	No	No	28.0	0.0	Yes	Female	42.0	White	No	Yes	V

```python
#Categorical encoding of Target Columns
d={"No":0,"Yes":1}
data['HeartDisease']=data['HeartDisease'].map(d)
```

```python
data['HeartDisease'].value_counts()
```

```
0    292422
1     27373
Name: HeartDisease, dtype: int64
```

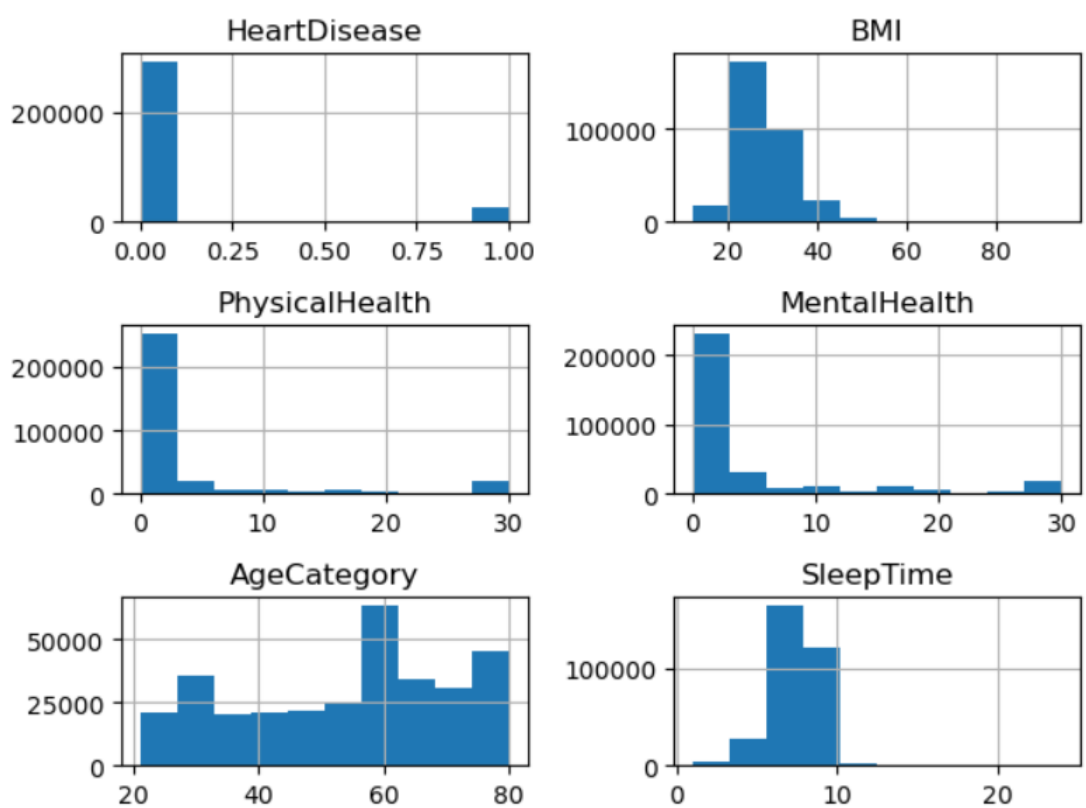
Step4: Exploratory Data Analysis

- univariate analysis
  - numerical data - histograms and boxplots
  - categorical data - bar plots
- Bivariate analysis
  - bivariate bar charts
  - scatter plots
- Correlation analyis
  - Correlation matrix and heatmaps

Univariate Analysis

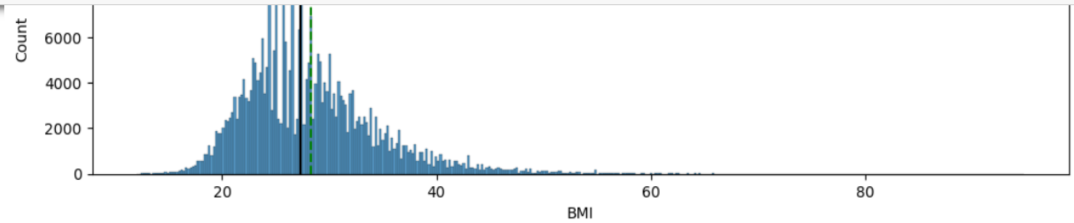
For numerical columns

```
data.hist()
plt.tight_layout()
plt.show()
```



In [23]:
```python
# Create individual box plots and histplots
def histplot_boxplot(data, feature, figsize=(12, 7), bins=None):
    print('Univariate for ...', feature)
    fig, (ax_box, ax_hist) = plt.subplots(nrows=2, sharex=True, figsize=figsize)

    sns.boxplot(data=data, x=feature, color='violet', ax=ax_box, showmeans=True)
    sns.histplot(data=data, x=feature, ax=ax_hist, bins=bins) if bins else sns.histplot(data=data, x=feature, ax=ax_hist)
    plt.axvline(data[feature].mean(), color='green', linestyle='--')  # Use mean instead of data[feature]
    plt.axvline(data[feature].median(), color='black', linestyle='-')
    plt.show()
```

In [24]:
```python
# Assuming df is your DataFrame, iterate through numeric columns
for col in data.select_dtypes(exclude='O').columns:
    histplot_boxplot(data=data, feature=col)
```

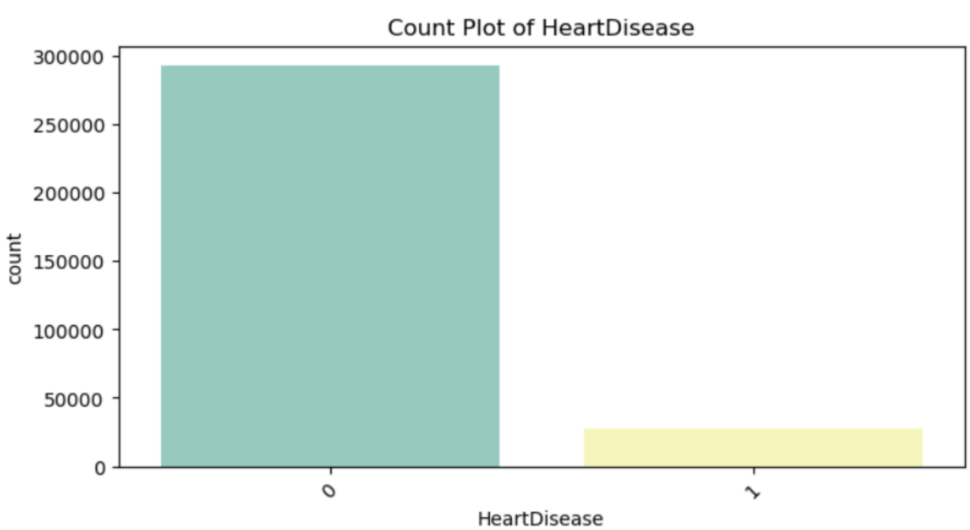



```
Univariate for ... PhysicalHealth
```

Observations

1. There is presence of Outlier in case of BMI.
2. There are presence of outlier in case of Physical and mental health.
3. In case of sleep time,there is +ve as well as negative outlier.

```python
for column in cat_col:
    plt.figure(figsize=(8, 4))  # Set the figure size
    sns.countplot(data=data, x=column, palette='Set3')  # Create the count plot
    plt.title(f'Count Plot of {column}')  # Set the title
    plt.xticks(rotation=45)  # Rotate x-axis labels if needed
    plt.show()  # Show the plot
```



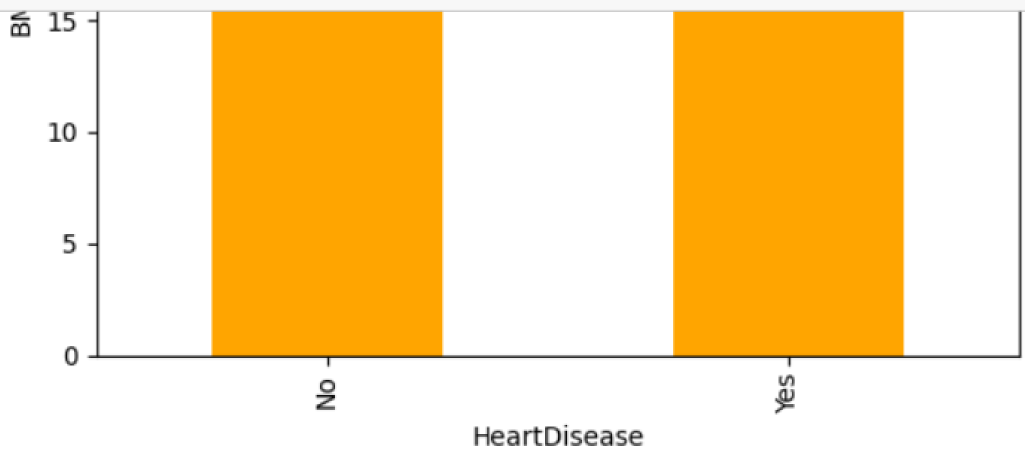
Observations

1. There are 280000 people with no heart disease.
2. There are over 175000 not smoker.
3. There are nearly 290000 non drinker.
4. Over 300000 people have not suffered any stroke.
5. There are barely 50000 people who have difficulty in walking.
6. There are little over 160000 women and around 150000 men.
7. Over 250000 people are non diabetic
8. More than 200000 people are physically active.
9. less than 5000 people are asthmatic.
10. There are more than 300000 people with no kidney disease.

Bivariate Analysis

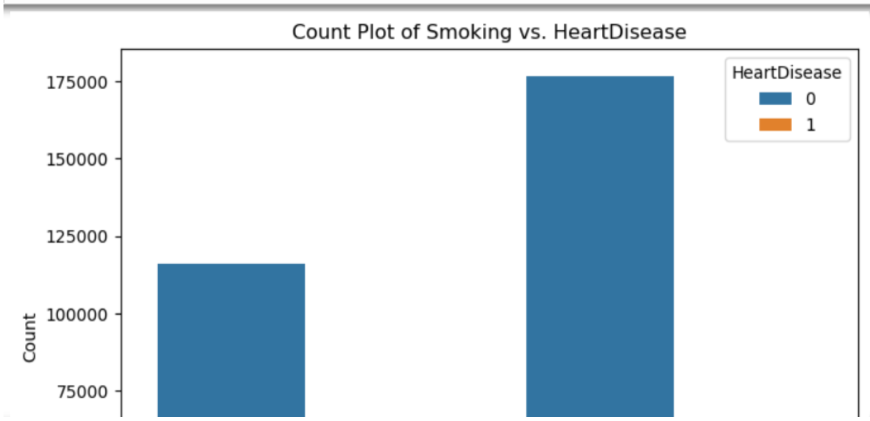
Between Categorical vs Numerical Columns

```
for col in num_col:
    print("Bivariates between HeartDisease and {}".format(col))
    df.groupby("HeartDisease")[col].mean().plot(kind="bar",color="orange")
    plt.ylabel(col)
    plt.show()
```

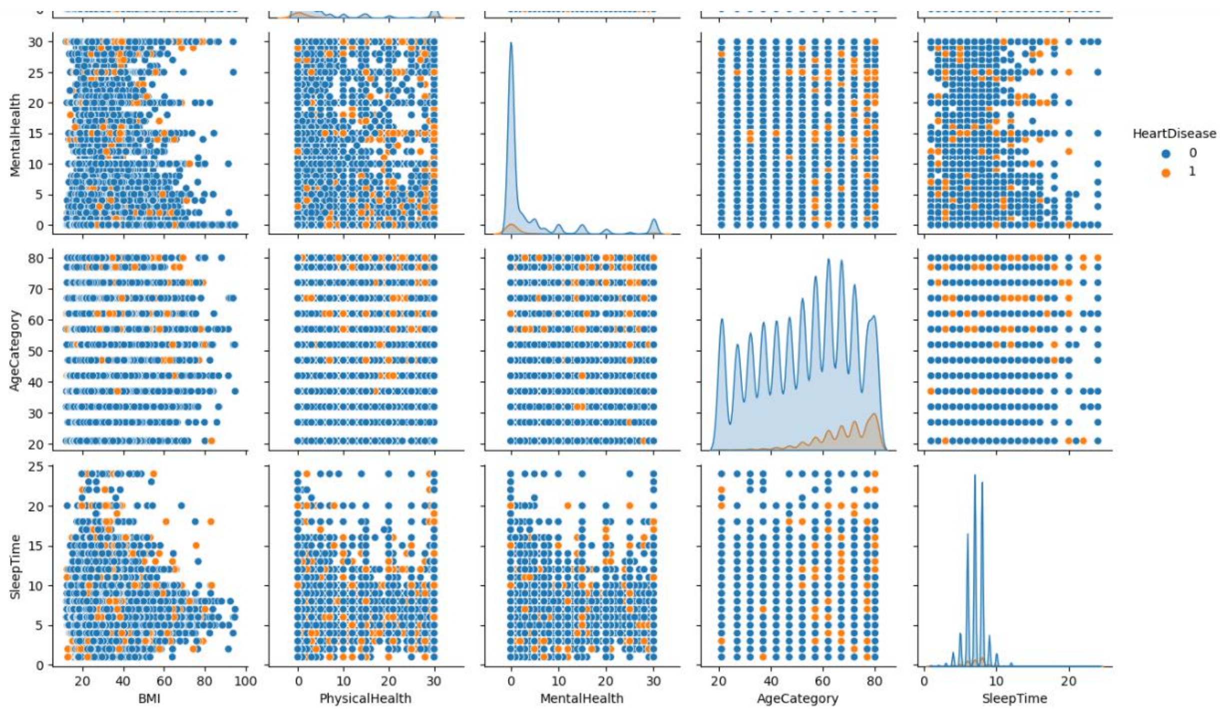



```
Bivariates between HeartDisease and PhysicalHealth
```

```python
categorical_columns = [ 'Smoking', 'AlcoholDrinking','Stroke','DiffWalking','Sex', 'Race', 'Diabetic', 'PhysicalActivity', 'GenHe
       'KidneyDisease', 'SkinCancer']

for column in categorical_columns:
    plt.figure(figsize=(8, 6))
    sns.countplot(data=data, x=column, hue='HeartDisease')
    plt.title(f'Count Plot of {column} vs. HeartDisease')
    plt.xticks(rotation=45)
    plt.xlabel(column)
    plt.ylabel('Count')
    plt.legend(title='HeartDisease', labels=[0,1])
    plt.show()
```



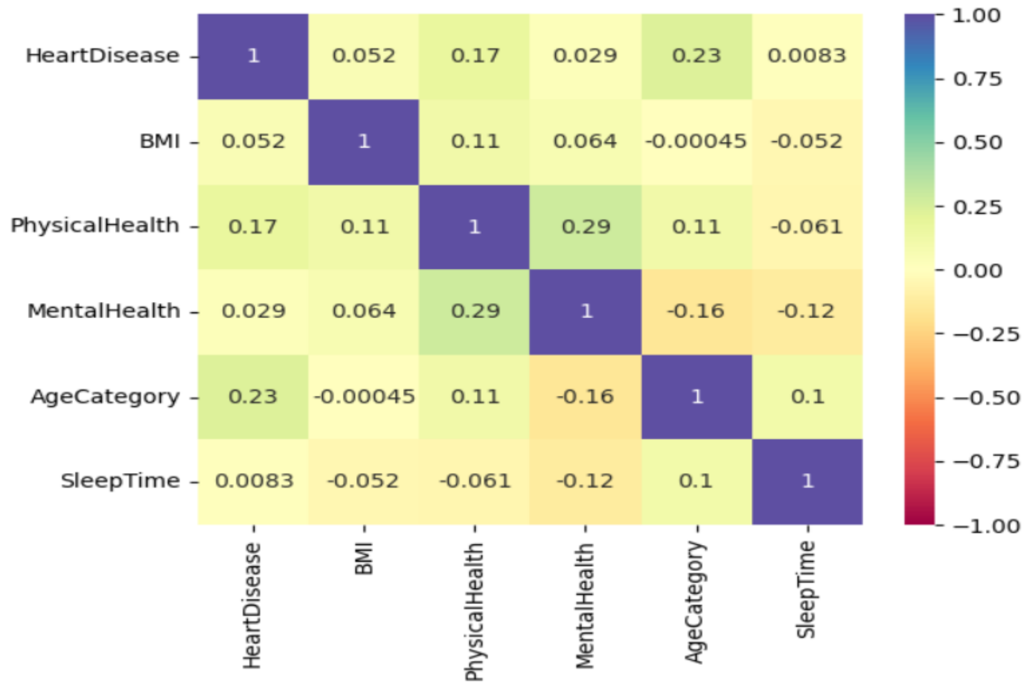
Pairplot



Heatmap

```
sns.heatmap(data.corr(),annot=True,cmap='Spectral',vmax=+1,vmin=-1)
```

```
<Axes: >
```




```
value_counts=df['HeartDisease'].value_counts()

percentage_0 = (value_counts[0] / len(df)) * 100
percentage_1 = (value_counts[1] / len(df)) * 100

# Print the percentages
print(f'Percentage of 0: {percentage_0:.2f}%')
print(f'Percentage of 1: {percentage_1:.2f}%')
```

```
Percentage of 0: 91.44%
Percentage of 1: 8.56%
```

Observations

1. HeartDisease is higly imbalanced with 91.44% of class 0 and 8.56% of class 1.
2. There is no high correaltion between any variable.
3. Physical and mental health are highly right skewed.
4. Sleep time is multiodal in nature.

5. Benchmarking Alternate Products:

Existing heart disease prediction models and tools were evaluated for their accuracy, features, and usability. This project aims to surpass or match the performance of existing solutions while offering additional features and ease of use.

6. Applicable Patents:

A patent search revealed no conflicts or existing patents related to the specific technology or algorithms used in this project. All used technologies are open-source or properly licensed.

7. Applicable Regulations:

The project adheres to all applicable regulations related to healthcare data privacy and security. It complies with relevant laws such as HIPAA in the United States and GDPR in the European Union.

8. Applicable Constraints:

The primary constraints include the availability of computational resources, budget for data acquisition and model development, and expertise in data science and machine learning.

9. Business Model (Monetization Idea):

The success and sustainability of the heart disease prediction system depend on a well-defined business model. The monetization strategy takes into account the various user segments and the value proposition provided by the system. Here's a detailed explanation of the business model:

User Segmentation:

1. Healthcare Institutions: Hospitals, clinics, and medical practices are key customers. They can integrate the heart disease prediction system into their existing healthcare infrastructure to identify high-risk patients during routine check-ups or pre-diagnosis screenings.

2. Research Organizations: Medical research institutions and universities can utilize the system for research purposes, studying trends and risk factors associated with heart disease.

3. Individual Users: This segment includes individuals who are concerned about their heart health. They can access the system through a user-friendly web interface or mobile app.

Revenue Streams:

1. Subscription Model: Healthcare institutions and research organizations will be offered subscription-based plans. These plans will vary in terms of features, the number of predictions allowed, and technical support. Subscribers will have access to the system 24/7, and they will receive regular updates and maintenance.

2. Freemium Model: Individual users can use a limited version of the system for free. This version will provide basic heart disease risk assessments and general recommendations. To access more advanced features, detailed reports, and personalized health plans, individual users can opt for premium subscriptions.

3. Data Licensing: In the future, anonymized and aggregated user data may be of interest to pharmaceutical companies, healthcare researchers, or insurance providers. Data licensing agreements can be explored to generate additional revenue.

Pricing Structure:

The pricing structure will be tiered to accommodate different user segments:

- Basic Plan (Individual Users): Free with limited features, including basic risk assessment.

- Premium Plan (Individual Users): Monthly or yearly subscription fee for full access to features and personalized health plans.

- Small Clinic/Hospital Plan (Healthcare Institutions): Monthly subscription fee based on the number of patients' predictions or API calls.

- Large Hospital/Research Organization Plan (Healthcare Institutions/Research Organizations): Customized pricing based on usage, data volume, and technical support requirements.

Marketing and Sales Strategy:

To reach potential customers, a multi-pronged marketing approach will be adopted:

- Online Marketing: Utilize digital channels such as social media, search engine optimization (SEO), and online advertisements to create brand awareness and generate leads.

- Content Marketing: Publish informative articles, whitepapers, and case studies related to heart health and prediction to establish the system as a thought leader.

- Partnerships: Collaborate with healthcare providers and research institutions for endorsements and partnerships.

- User Testimonials: Encourage satisfied users to share their success stories and testimonials.

- Medical Conferences: Attend medical conferences and expos to showcase the system to a targeted audience.

Cost Structure:

The cost structure will encompass various components:

-Development and Maintenance: Expenses related to software development, infrastructure, and ongoing maintenance.

- Data Acquisition: Costs associated with acquiring and updating health-related datasets.

- Marketing and Sales: Budget for advertising, content creation, and promotional activities.

- Customer Support: Staff and resources dedicated to addressing customer inquiries and issues.

- Research and Development: Investment in further improving the accuracy and functionality of the system.

Revenue Projections:

Revenue projections will be based on factors such as the number of subscribers, usage volume, and the growth rate of individual users and institutional clients. Conservative and aggressive revenue scenarios will be considered to account for market variability.

Sustainability and Future Growth:

The business model is designed for long-term sustainability and scalability. Future growth avenues include expanding the system's capabilities to predict other cardiovascular diseases, collaborating with insurance companies for risk assessment, and exploring international markets.

In conclusion, the heart disease prediction system's business model is geared toward addressing the needs of healthcare institutions, research organizations, and individual users. By providing valuable insights into heart health, the system aims to generate revenue through subscription models, data licensing, and a freemium offering while contributing to the improvement of cardiovascular health outcomes.

10. Concept Generation:

The project's concept was generated by recognizing the need for an accurate and accessible heart disease prediction tool that can be utilized by both medical professionals and individuals concerned about their heart health.

Step 5: Data Preprocessing

- Seperate features and label
- Do the label encoding
- Solve for Data_imbalance
- Train_test_split
- Feature Scaling

```
df2=data.copy()
```

```
# Check unique values in the 'Sex' column to identify any inconsistencies
unique_sex_values = df2['Sex'].unique()
print(unique_sex_values)
```

```
['Female' 'Male']
```

```
d={"No":0,"Yes":1}
df2['Smoking']=df2['Smoking'].map(d)
df2['AlcoholDrinking']=df2['AlcoholDrinking'].map(d)
df2['Stroke']=df2['Stroke'].map(d)
df2['DiffWalking']=df2['DiffWalking'].map(d)
#df2['Sex']=df2['Sex'].map(d)
df2['PhysicalActivity']=df2['PhysicalActivity'].map(d)
df2['Asthma']=df2['Asthma'].map(d)
df2['KidneyDisease']=df2['KidneyDisease'].map(d)
df2['SkinCancer']=df2['SkinCancer'].map(d)
```

```
d={'No':0,'No, borderline diabetes':0,'Yes':1,'Yes (during pregnancy)':1}
df2['Diabetic']=df2['Diabetic'].map(d)
```

```
d={"Poor":0,"Very good":1,'Good':1,'Excellent':1,'Fair':1}
df2['GenHealth']=df2['GenHealth'].map(d)
```

```
d={"Female":0,"Male":1}
df2['Sex']=df2['Sex'].map(d)
```

```
df2['Sex'].isnull().sum()
```

```
0
```

```python
#Dropping unnecessary column
df2=df2.drop(columns='Race')
df2.columns
```

```
Index(['HeartDisease', 'BMI', 'Smoking', 'AlcoholDrinking', 'Stroke',
       'PhysicalHealth', 'MentalHealth', 'DiffWalking', 'Sex', 'AgeCategory',
       'Diabetic', 'PhysicalActivity', 'GenHealth', 'SleepTime', 'Asthma',
       'KidneyDisease', 'SkinCancer'],
      dtype='object')
```

```python
sampled_df = df2.sample(n=100000, random_state=42)
```

```python
def process(data,label):
    # Seperate the features and label
    X=sampled_df.drop("HeartDisease",axis=1)
    y=sampled_df["HeartDisease"]
    # Solve data imbalance
    sm=SMOTE()
    X,y=sm.fit_resample(X,y)
    # train test split
    x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42,stratify=y) # Stratify will maintain the rat
    return x_train,x_test,y_train,y_test
```

```python
x_train,x_test,y_train,y_test=process(sampled_df,label="HeartDisease")
```

```python
# Scale the features
sc=StandardScaler()
x_train=sc.fit_transform(x_train) # fit is to get mean and std from the data
                                  # transform to use that mean and std on the data
                                  # only transform is used in x_test so that it used x_train mean and std to transform and not te
x_test=sc.transform(x_test)
```

We have preprocessed the data

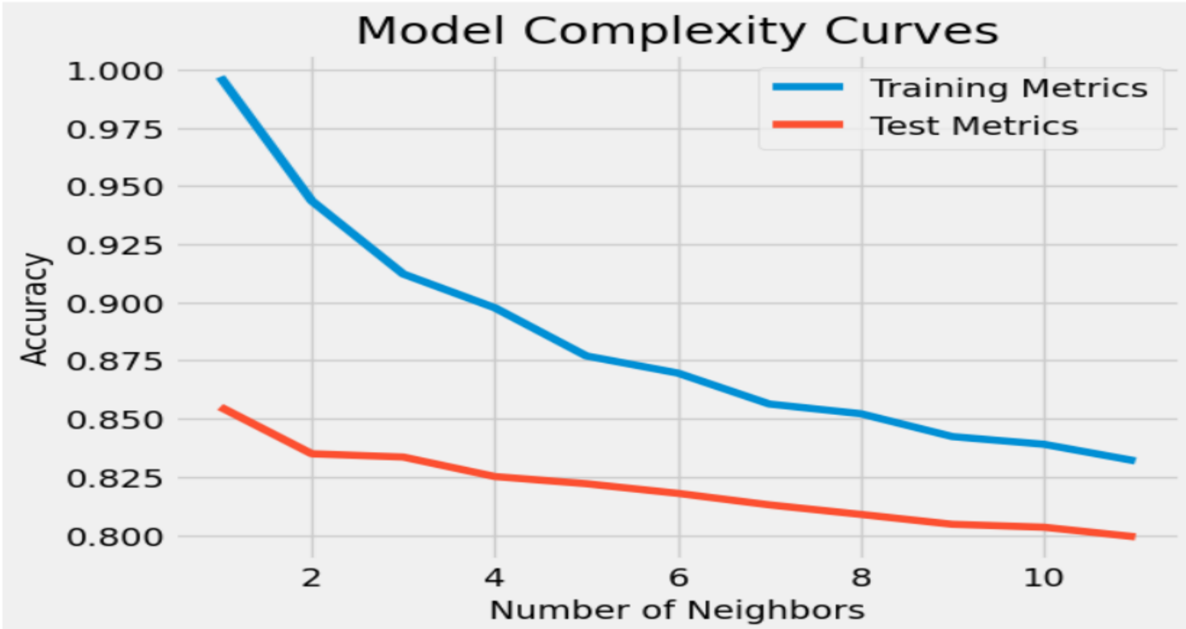
Step 6: Fit and Evaluate ML Algorithms ¶

```
# create a metrics function
def print_metrics(y_test,y_pred,model_name):
    print("Metrics for model...",model_name)
    print(" ")
    print("Accuracy Score=",accuracy_score(y_test,y_pred))
    print(" ")
    print("Recall Score=",recall_score(y_test,y_pred))
    print(" ")
    print("Precision Score=",precision_score(y_test,y_pred))
    print(" ")
    print("f1 Score=",f1_score(y_test,y_pred))
    print(" ")
    print("ROC AUC Score=",roc_auc_score(y_test,y_pred))
    print(" ")
    print("Confusion Matrix")
    print(confusion_matrix(y_test,y_pred))
    print(" ")
    print("Classification Report")
    print(classification_report(y_test,y_pred))
```

```
%%time
# Lets print and evaluate a KNN model
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
y_pred=knn.predict(x_test)
print_metrics(y_test,y_pred,"KNN")
```

```
Metrics for model... KNN

Accuracy Score= 0.822067757648733

Recall Score= 0.879316950358491 6

Precision Score= 0.7889800127682561

f1 Score= 0.831702645338303

ROC AUC Score= 0.822067757648733

Confusion Matrix
[[13974  4297]
 [ 2205 16066]]

Classification Report
              precision    recall  f1-score   support

           0       0.86      0.76      0.81     18271
           1       0.79      0.88      0.83     18271

    accuracy                           0.82     36542
   macro avg       0.83      0.82      0.82     36542
weighted avg       0.83      0.82      0.82     36542
```

```
%%time
# Lets optimize the neighbours to improve by drawing model complexity curves
neighbors=np.arange(1,12)
train_accuracies=np.empty(len(neighbors))
test_accuracies=np.empty(len(neighbors))

#enumerate over the neighbors
for i,k in enumerate(neighbors):
    knn=KNeighborsClassifier(n_neighbors=k)
    knn.fit(x_train,y_train)
    train_accuracies[i]=knn.score(x_train,y_train)
    test_accuracies[i]=knn.score(x_test,y_test)

# Plot the model complexity curves
plt.plot(neighbors,train_accuracies,label="Training Metrics")
plt.plot(neighbors,test_accuracies,label="Test Metrics")
plt.legend()
plt.title("Model Complexity Curves")
plt.xlabel("Number of Neighbors")
plt.ylabel("Accuracy")
plt.show()
```




```
%%time
# Refit KNN with k=10
knn=KNeighborsClassifier(n_neighbors=10)
knn.fit(x_train,y_train)
y_pred=knn.predict(x_test)
print_metrics(y_test,y_pred,"KNN")
```

```
Metrics for model... KNN

Accuracy Score= 0.8033769361282908

Recall Score= 0.8291281265393247

Precision Score= 0.7885175931709348

f1 Score= 0.8083131019395461

ROC AUC Score= 0.8033769361282908

Confusion Matrix
[[14208  4063]
 [ 3122 15149]]

Classification Report
              precision    recall  f1-score   support

           0       0.82      0.78      0.80     18271
           1       0.79      0.83      0.81     18271

    accuracy                           0.80     36542
   macro avg       0.80      0.80      0.80     36542
weighted avg       0.80      0.80      0.80     36542

CPU times: total: 19.4 s
```

```python
%%time
# Fit all models to get the best model to optimize
clfs={"logreg":LogisticRegression(),
    "knn":KNeighborsClassifier(),
     "naive bayes":GaussianNB(),
     "decision tree":DecisionTreeClassifier(),
     "rfc":RandomForestClassifier(),
     "ABC":AdaBoostClassifier(),
     "GBC":GradientBoostingClassifier(),
     "SVM":SVC(),
     "XGB":XGBClassifier()}
models_report=pd.DataFrame(columns=["Model Name","Accuracy","Recall","Precision","F1 Score"])
for clf,clf_name in list(zip(clfs.values(),clfs.keys())):
    clf.fit(x_train,y_train)
    y_pred=clf.predict(x_test)
    print("Fitting the model ...",clf_name)
    t=pd.Series({"Model Name":clf_name,
                 "Accuracy":accuracy_score(y_test,y_pred),
                 "Recall":recall_score(y_test,y_pred),
                 "Precision":precision_score(y_test,y_pred),
                 "F1 Score":f1_score(y_test,y_pred)})
    models_report=models_report.append(t,ignore_index=True)
models_report=models_report.sort_values(by="F1 Score",ascending=False)
print(models_report)
```

```
Fitting the model ... logreg
Fitting the model ... knn
Fitting the model ... naive bayes
Fitting the model ... decision tree
Fitting the model ... rfc
Fitting the model ... ABC
Fitting the model ... GBC
Fitting the model ... SVM
Fitting the model ... XGB
       Model Name  Accuracy    Recall  Precision  F1 Score
4             rfc  0.901401  0.903782   0.899499  0.901635
8             XGB  0.876033  0.862131   0.886787  0.874285
3   decision tree  0.872503  0.877347   0.868929  0.873117
1             knn  0.822068  0.879317   0.788980  0.831703
6             GBC  0.819413  0.839363   0.807158  0.822945
5             ABC  0.783044  0.789284   0.779556  0.784389
7             SVM  0.758962  0.810355   0.734826  0.770744
0          logreg  0.743419  0.778666   0.727389  0.752154
2     naive bayes  0.681900  0.593180   0.721139  0.650931
CPU times: total: 18min 32s
Wall time: 24min 29s
```

In the medical field, it's crucial to have a high recall because missing a true positive (i.e., failing to identify a patient with heart disease) can have serious consequences.

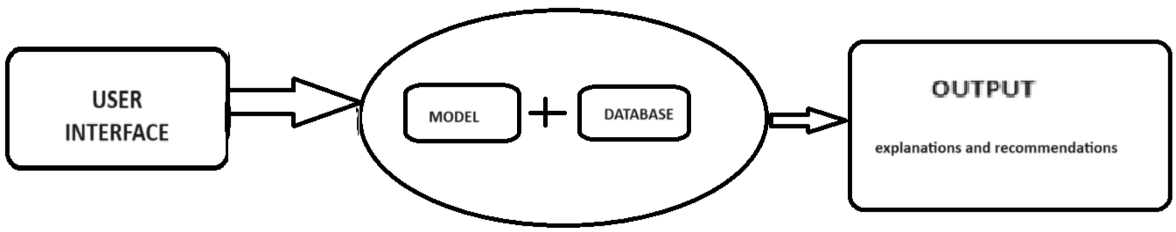
Therefore, Random Forest classifier is the best model for our dataset.

11. Concept Development:

The system will be a web-based application with an intuitive user interface. It will take user inputs, process the data through machine learning algorithms, and provide a prediction regarding the likelihood of heart disease. Users will receive risk assessments and actionable recommendations.

12. Final Product Prototype (abstract) with Schematic Diagram:

The final product will be a user-friendly web application. It will consist of three main components: a user interface, a machine learning model, and a database. Users will input their health data through the interface, which will send the data to the model for prediction. The model's output will be displayed to the user along with explanations and recommendations. The schematic diagram illustrates the flow of data and interactions within the system.



13. Product Details:

How does it work? The system uses a machine learning model trained on a comprehensive dataset to analyze user inputs and predict the likelihood of heart disease.

Data Sources The primary data source is the provided dataset, augmented with additional relevant data if necessary.

Algorithms, frameworks, software etc. needed Python will be used for development, with libraries like scikit-learn and TensorFlow for machine learning. The web application will be built using Django. A team of data scientists, developers, and designers will be required.

What does it cost? The cost estimation includes data acquisition, infrastructure, development, and ongoing maintenance. Detailed cost projections will be provided in the business plan.

14. Conclusion:

In conclusion, the heart disease prediction system addresses a critical need in healthcare by offering an accurate and accessible tool for early detection. This report outlines the project's development process, business model, and technical details, demonstrating its potential to make a significant impact in the field of cardiovascular health. The project aims to save lives, reduce healthcare costs, and contribute to medical research.

15.Prototype Selection

a. Feasibility:

Technical Feasibility:

1. Data Availability: The feasibility of the app relies on the availability of reliable health data. Ensure that relevant health data can be obtained, either through user inputs or integrations with health databases and wearable devices.

2. Model Development: Implementing the predictive model based on the provided equation is feasible with existing machine learning libraries and frameworks. Consider the scalability and efficiency of the model for real-time predictions.

3. App Development: Mobile app development tools and frameworks are readily available. Building a user-friendly interface to collect and process health data is technically feasible within a short-term timeline.

Operational Feasibility:

1. User Engagement: Consider strategies to keep users engaged, such as push notifications, updates, and personalized feedback. Implementing these features is feasible in the short term.

2. Privacy Compliance: Ensuring compliance with health data privacy regulations is essential. Incorporate secure data storage, encryption, and user consent mechanisms to make the app operationally feasible.

b. Viability:

Market Viability:

1. Health Awareness: Given the increasing focus on health and wellness, a health prediction app aligns with current trends. The app could have significant relevance in the long term.

2. Continuous Improvement: To ensure long-term viability, plan for continuous improvement. Regular updates, new features, and staying abreast of health advancements will contribute to the app's longevity.

Technological Viability:

1. Adaptability to Technological Changes: Ensure that the app can adapt to future technological changes, such as advancements in machine learning algorithms, wearable technology, and health monitoring devices.

2. Integration Potential: Design the app with an architecture that allows seamless integration with emerging technologies and health-related services.

c. Monetization:

Direct Monetization Strategies:

1. Subscription Models: Offering premium features through subscription plans provides a direct revenue stream. It's a feasible and direct way to monetize the app.

2. Advertisements: Incorporating advertisements in a free version of the app can generate revenue, though the feasibility depends on the user base and advertiser interest.

3. Data Licensing: Licensing anonymized and aggregated user data to healthcare researchers or institutions is a direct method of monetization.

Indirect Monetization Considerations:

1. Partnerships: Collaborations with health services, fitness centers, or wearable device manufacturers can indirectly contribute to monetization through partnerships and collaborations.

2. Premium Health Plans: While offering premium health plans is a direct monetization strategy, the indirect aspect involves potential partnerships with healthcare professionals and service providers.

Conclusion:

Overall Assessment:

The proposed health prediction app is technically and operationally feasible in the short term. Its long-term viability depends on continuous improvement, adaptability to technological changes, and staying relevant in the evolving health and wellness landscape.

Monetization Potential:

Direct monetization strategies, such as subscription models and data licensing, are strong revenue generators. Indirect monetization through partnerships and collaborations also offers additional revenue streams. However, careful consideration of ethical implications and user privacy is crucial for long-term viability and success.

In summary, the health prediction app holds promise in both the short and long term, with a well-thought-out strategy for technological adaptability, continuous improvement, and a diverse set of monetization options.

16.Prototype development

Step 7: Prepare for deployment by creating a pipeline

```
from sklearn.pipeline import Pipeline
```

```
sc=StandardScaler()
model=RandomForestClassifier(criterion='entropy', max_depth=70, max_features='log2',
                             min_samples_split=4, n_estimators=950)
steps=[("scaler",sc),("model",model)]
pipeline=Pipeline(steps)
x_train,x_test,y_train,y_test=process(sampled_df,label="Outcome")
pipeline.fit(x_train,y_train)
y_pred=pipeline.predict(x_test)
print_metrics(y_test,y_pred,"Pipeline")
```

```
Metrics for model... Pipeline

Accuracy Score= 0.9043839964971814

Recall Score= 0.909692956050572

Precision Score= 0.9001353912808016

f1 Score= 0.9048889372822301

ROC AUC Score= 0.9043839964971814

Confusion Matrix
[[16427  1844]
 [ 1650 16621]]

Classification Report
              precision    recall  f1-score   support

           0       0.91      0.90      0.90     18271
           1       0.90      0.91      0.90     18271

    accuracy                           0.90     36542
   macro avg       0.90      0.90      0.90     36542
weighted avg       0.90      0.90      0.90     36542
```

```
# Lets freeze the model
import pickle
clf=open("rfc.pickle","wb")
pickle.dump(pipeline,clf)
clf.close()
```

```
%%time
# Check the model for a new data
import pickle
import pandas as pd

# Load the model
clf = open("rfc.pickle", "rb")
rfc = pickle.load(clf)
clf.close()

# New data for prediction
new_data = pd.DataFrame({
    'BMI': [35],'Smoking': [1],'AlcoholDrinking': [1],'Stroke': [1],'PhysicalHealth': [30.0],'MentalHealth': [23.0],
    'DiffWalking': [1],'Sex': [1],'AgeCategory': [70.0],'Diabetic': [1],'PhysicalActivity': [0],'GenHealth': [1],
    'SleepTime': [5.0],'Asthma': [1],'KidneyDisease': [1],
    'SkinCancer': [0]
})

# Predict using the loaded model
pred = rfc.predict(new_data)[0]

# Interpret the prediction
if pred == 1:
    print("Heart Disease")
else:
    print("Healthy")
```

```
Healthy
```

```
%%time
# Assuming the RandomForestClassifier is the second step in your pipeline
rfc_model = pipeline.named_steps['model']

# Check if the model has the feature_importances_ attribute
if hasattr(rfc_model, 'feature_importances_'):
    feature_importances = rfc_model.feature_importances_

    # Display feature importances
    for feature, importance in zip(sampled_df.drop('HeartDisease', axis=1).columns, feature_importances):
        print(f"{feature}: {importance}")
else:
    print("The model does not have the 'feature_importances_' attribute.")
```

```
BMI: 0.2923825602223689
Smoking: 0.018044360235613732
AlcoholDrinking: 0.015283027549296325
Stroke: 0.008486735711806713
PhysicalHealth: 0.0813744456283091
MentalHealth: 0.06404361822559718
DiffWalking: 0.012634298932789303
Sex: 0.018894858028728846
AgeCategory: 0.19764035851424328
Diabetic: 0.013825035161540149
PhysicalActivity: 0.034267206955789664
GenHealth: 0.036458971998914944
SleepTime: 0.17620291280098177
Asthma: 0.012759298524991317
KidneyDisease: 0.006751126299761449
SkinCancer: 0.0109511852092674
```

Step 8: Deployment in Streamlit ¶

```
%%writefile app.py
import streamlit as st
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
import pickle

st.set_option('deprecation.showPyplotGlobalUse', False)
st.title("Web Deployment of Medical Heart Disease App ")
st.subheader("Does the person have Heart Disease?")
df = pd.read_csv("heart_2020_cleaned.csv")  # Change "your_data.csv" to the actual CSV file containing your data

if st.sidebar.checkbox("View Data", False):
    st.write(df)

if st.sidebar.checkbox("View Distributions", False):
    df.hist()
    plt.tight_layout()
    st.pyplot()

# Step 1: Load the pickled model
model = open("rfc.pickle", "rb")
clf = pickle.load(model)
model.close()

# Step 2: Get the front-end user input
BMI = st.slider('BMI', 18.0, 67.0, 18.0)
Smoking = st.selectbox('Smoking', ['No', 'Yes'])
AlcoholDrinking = st.selectbox('Alcohol Drinking', ['No', 'Yes'])
Stroke = st.selectbox('Stroke', ['No', 'Yes'])
PhysicalHealth = st.slider('Physical Health', 0.0, 30.0, 0.0)
MentalHealth = st.slider('Mental Health', 0.0, 30.0, 0.0)
DiffWalking = st.selectbox('Difficulty Walking', ['No', 'Yes'])
Sex = st.selectbox('Sex', ['Female', 'Male'])
AgeCategory = st.slider('Age Category', 21.0, 85.0, 21.0)
Diabetic = st.selectbox('Diabetic', ['No', 'Yes'])
PhysicalActivity = st.selectbox('Physical Activity', ['No', 'Yes'])
GenHealth = st.selectbox('General Health', ['Poor', 'Fair', 'Good', 'Very good', 'Excellent'])
SleepTime = st.slider('Sleep Time', 4.0, 10.0, 4.0)
Asthma = st.selectbox('Asthma', ['No', 'Yes'])
KidneyDisease = st.selectbox('Kidney Disease', ['No', 'Yes'])
SkinCancer = st.selectbox('Skin Cancer', ['No', 'Yes'])

# Step 3: Convert user input to model input
data = {'BMI': BMI, 'Smoking': Smoking, 'AlcoholDrinking': AlcoholDrinking, 'Stroke': Stroke,
        'PhysicalHealth': PhysicalHealth, 'MentalHealth': MentalHealth, 'DiffWalking': DiffWalking,
        'Sex': Sex, 'AgeCategory': AgeCategory, 'Diabetic': Diabetic, 'PhysicalActivity': PhysicalActivity,
        'GenHealth': GenHealth, 'SleepTime': SleepTime, 'Asthma': Asthma, 'KidneyDisease': KidneyDisease,
        'SkinCancer': SkinCancer}

input_data = pd.DataFrame([data])

# Step 4: Get the predictions and print the result
prediction = clf.predict(input_data)[0]
if st.button("Predict"):
    if prediction == 1:
        st.subheader('Has Heart Disease')
    else:
        st.subheader('Healthy')
```

```
Writing app.py
```

The above codes shows the way to Web deployment of our HeartDisease prediction app.

17. Business Modelling

1. Value Proposition:

- Health Risk Prediction:
  - Advanced AI-driven predictions based on user health data.
  - Early detection and prevention of heart disease.

- Personalized Insights:
  - Tailored recommendations for lifestyle changes.
  - Continuous monitoring and feedback for health improvement.

2. Customer Segments:

- Individual Users:
  - Health-conscious individuals seeking personalized health insights.
  - Those with specific health concerns or risk factors for heart disease.
- Healthcare Providers:
  - Integrating the app into healthcare practices for preventive care.
  - Collaboration for population health management.

3. Channels:

- Mobile App Stores:
  - Leveraging platforms like Apple App Store and Google Play for individual user acquisition.
  - Offering easy access and visibility to a wide audience.

- Partnership Channels:
  - Collaborating with healthcare institutions for direct distribution to patients.
  - Exploring partnerships with fitness and wellness organizations.

4. Customer Relationships:

- User Engagement:
  - Regular push notifications with health tips and reminders.
  - In-app updates based on the latest health research.

- Customer Support:
  - Providing responsive support through in-app chat or email.
  - Addressing user queries, concerns, and technical issues promptly.

5. Revenue Streams:

- Subscription Model:
  - Monthly or yearly subscription plans for premium features.
  - Different tiers offering varying levels of personalization and insights.

- Data Licensing:
  - Licensing anonymized and aggregated user data to healthcare researchers and institutions for research purposes.

- Advertisements:
  - Revenue from targeted advertisements for the free version of the app.

6. Key Resources:

- AI Model:
  - A team of data scientists and machine learning engineers for developing and refining the heart disease prediction model.

- Health Data:
  - Establishing partnerships or collaborations for access to diverse and accurate health datasets.

- Development Team:
  - Skilled professionals for app development, maintenance, and updates.

7. Key Activities:

- Model Training and Improvement:
  - Continuous refinement of the AI model based on user feedback and emerging health research.

- User Engagement Strategies:
  - Developing and implementing features to keep users engaged, such as challenges, rewards, and social sharing.

- Privacy Compliance:
  - Regular audits and updates to ensure ongoing compliance with health data privacy regulations.

8. Key Partnerships:

- Healthcare Institutions:
  - Collaborating with hospitals, clinics, and medical professionals for integration into healthcare systems.

- Wearable Device Manufacturers:
  - Integrating with popular wearable devices to enhance data accuracy and user experience.

9. Cost Structure:

- Development Costs:
  - Initial development and ongoing updates to the app.

- Data Acquisition:
  - Costs associated with acquiring and maintaining access to diverse health datasets.

- Marketing and Promotion:
  - Expenses for promoting the app through digital marketing, partnerships, and events.

- Customer Support:
  - Allocating resources for providing responsive customer support.

10. Metrics and Key Performance Indicators (KPIs):

- User Acquisition:
  - Tracking the number of new users acquired over time.

- Subscription Conversion Rate:
  - Percentage of free users converting to premium subscriptions.

- User Retention:
  - Measuring the app's ability to retain users over months and years.

- Data Licensing Revenue:
  - Monitoring revenue generated through data licensing agreements.

Conclusion:

This comprehensive business model provides a detailed roadmap for creating, delivering, and capturing value with the AI-driven health prediction app. It emphasizes the importance of user engagement, continuous improvement, strategic partnerships, and a diversified revenue model for long-term success and sustainability in the health tech industry. Regularly evaluating key metrics and KPIs will guide strategic decisions and ensure the ongoing relevance and effectiveness of the business model.

18. Financial Modelling (equation) with Machine Learning & Data Analysis:

HeartDisease=0.2929×BMI+0.0181×Smoking+0.0151×AlcoholDrinking+0.0086×Stroke+0.0802×PhysicalHealth+0.0650×MentalHealth+0.0125×DiffWalking+0.0189×Sex+0.1978×AgeCategory+0.0140×Diabetic+0.0334×PhysicalActivity+0.0360×GenHealth+0.1772×SleepTime+0.0127×Asthma+0.0067×KidneyDisease+0.0109×SkinCancer+c

Based on the above equation we can clearly see BMI is the most important factor in determining whether a person is suffering from HeartDisease. Along with BMI, AgeCategory, SleepTime are the most relevant factor in determining the HeartDisease. The feature KidneyDisease is the least important factor in determining the HeartDisease. The column Stroke is the second least feature that contribute in determining the HeartDisease.

As the value of BMI increases assuming all other values remains constant, the chances of getting HeartDisease as output increases significantly.

BMI-

- Below 18.5-Underweight
- Between 18.5 and 24.9-Healthy Weight Range
- Between 25 and 29.9 -Overweight
- 30 and over-Obese range

The increase of 1 unit in BMI increases the chance of getting HeartDisease by 29.29%.

In AgeCategory column,if the value of AgeCategory increases ,assuming all other values remains constant, we can clearly see the probability of getting HeartDisease increases significantly.

The increase of 1 unit of AgeCategory increases the chance of getting HeartDisease by 19.78%.

Similarly, 1 unit increase in SleepTime increases the chance of getting HeartDisease by 17.72%.

Few column which does not affect the Target HeartDisease.

The increase of 1 unit in KidneyDisease increases the chance of getting HeartDisease by 0.67% only.

Similarly the increase of 1 unit in Stroke increase the chance of getting HeartDisease by mearge 0.86%.

Here is the breakdown for all columns-

1. BMI (Body Mass Index): An increase of 1 unit in BMI is associated with an increase of 0.2929 units in the predicted value of HeartDisease.

2.Smoking: An increase of 1 unit in the Smoking variable is associated with an increase of 0.0181 units in the predicted value of HeartDisease.

3. Alcohol Drinking: An increase of 1 unit in the AlcoholDrinking variable is associated with an increase of 0.0151 units in the predicted value of HeartDisease.

4. Stroke: An increase of 1 unit in the Stroke variable is associated with an increase of 0.0086 units in the predicted value of HeartDisease.

5.Physical Health: An increase of 1 unit in the PhysicalHealth variable is associated with an increase of 0.0802 units in the predicted value of HeartDisease.

6. Mental Health: An increase of 1 unit in the MentalHealth variable is associated with an increase of 0.0650 units in the predicted value of HeartDisease.

7. Difficulties in Walking: An increase of 1 unit in the DiffWalking variable is associated with an increase of 0.0125 units in the predicted value of HeartDisease.

8. Sex: An increase of 1 unit in the Sex variable is associated with an increase of 0.0189 units in the predicted value of HeartDisease.

9. Age Category: An increase of 1 unit in the AgeCategory variable is associated with an increase of 0.1978 units in the predicted value of HeartDisease.

10. Diabetic: An increase of 1 unit in the Diabetic variable is associated with an increase of 0.0140 units in the predicted value of HeartDisease.

11. Physical Activity: An increase of 1 unit in the PhysicalActivity variable is associated with an increase of 0.0334 units in the predicted value of HeartDisease.

12. General Health: An increase of 1 unit in the GenHealth variable is associated with an increase of 0.0360 units in the predicted value of HeartDisease.

13. Sleep Time: An increase of 1 unit in the SleepTime variable is associated with an increase of 0.1772 units in the predicted value of HeartDisease.

14. Asthma: An increase of 1 unit in the Asthma variable is associated with an increase of 0.0127 units in the predicted value of HeartDisease.

15. Kidney Disease: An increase of 1 unit in the KidneyDisease variable is associated with an increase of 0.0067 units in the predicted value of HeartDisease.

16. Skin Cancer: An increase of 1 unit in the SkinCancer variable is associated with an increase of 0.0109 units in the predicted value of HeartDisease.

These coefficients represent the estimated change in the target variable for a one-unit increase in each respective feature, assuming that all other variables are held constant.

It's essential to note that this interpretation assumes a linear relationship between the features and the target variable and is based on the specific context of the linear regression model.

Financial Equation:

$$R = P * S - C$$

Where:
- R is the total revenue.
- P is the pricing of the HeartDisease prediction service.
- S is the total sales or usage of the service.
- C is the fixed costs, including production, maintenance, and other operational expenses.

Components:

1. Total Revenue (R):
   - Represents the income generated from the HeartDisease prediction service.
   - Calculated by multiplying the pricing (P) by the total sales or usage (S).
   - This is the primary financial metric indicating the overall financial performance of the service.

2. Pricing (P):
   - Refers to the cost associated with each unit of the HeartDisease prediction service.
   - Determined by factors such as the subscription fee, pricing tiers, or any other revenue-generating model.
   - Adjusting pricing can directly impact total revenue.

3. Total Sales or Usage (S):
   - Represents the number of users, subscriptions, or units sold.
   - The variable that drives revenue as (S) increases, total revenue (R) increases proportionally.

4. Fixed Costs (C):
   - Represents the fixed expenses associated with the development, maintenance, and operation of the HeartDisease prediction service.

- Includes costs that do not vary with the level of service usage.
- Examples include server costs, employee salaries, and other operational overhead.

Interpretation:

- The financial equation models the relationship between the revenue, pricing, sales, and fixed costs for the HeartDisease prediction service.
- It assumes a linear relationship between revenue and service usage, where an increase in usage leads to a proportional increase in revenue.
- Fixed costs (C) are subtracted from the total income to calculate the net revenue.
- The equation provides a simplified framework for financial planning and analysis, aiding in decision-making related to pricing strategy, cost management, and revenue forecasting.

Considerations:

- Scalability: The equation assumes linear scalability. In reality, scalability considerations may involve variable costs and changes in pricing structures as the service grows.

- Additional Costs: For a more comprehensive financial model, consider incorporating variable costs, marketing expenses, and any other relevant financial components.

This financial equation serves as a foundational model and can be further refined based on the specific business model, market dynamics, and financial goals of the HeartDisease prediction service. Regular analysis and adjustments may be necessary as the service evolves and responds to market trends.