

# **Heart Disease Prediction**

**By Hrithik**

**Dataset Source-**

**<https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease>**

## **Objective**

To predict whether the respondents will ever have coronary heart disease (CHD) or myocardial infarction (MI)

## **Description of the Data Set**

According to the [CDC](#), heart disease is one of the leading causes of death for people of most races in the US (African Americans, American Indians and Alaska Natives, and white people). About half of all Americans (47%) have at least 1 of 3 key risk factors for heart disease: high blood pressure, high cholesterol, and smoking. Other key indicators include diabetic status, obesity (high BMI), not getting enough physical activity or drinking too much alcohol. Detecting and preventing the factors that have the greatest impact on heart disease is very important in healthcare. Computational developments, in turn, allow the application of machine learning methods to detect "patterns" from the data that can predict a patient's condition.

## Data Description

- HeartDisease: Respondents that have ever reported having coronary heart disease (CHD) or myocardial infarction (MI).
- BMI: Body Mass Index (BMI).
- Smoking: Have you smoked at least 100 cigarettes in your entire life?
- AlcoholDrinking: Heavy drinkers (adult men having more than 14 drinks per week and adult women having more than 7 drinks per week)
- Stroke: (Ever told) (you had) a stroke?
- PhysicalHealth: Now thinking about your physical health, which includes physical illness and injury, for how many days during the past 30 days was your physical health not good? (0-30 days).
- MentalHealth: Thinking about your mental health, for how many days during the past 30 days was your mental health not good? (0-30 days).
- DiffWalking: Do you have serious difficulty walking or climbing stairs?
- Sex: Are you male or female?
- AgeCategory: Fourteen-level age category. (then calculated the mean)
- Race: Imputed race/ethnicity value.
- Diabetic: (Ever told) (you had) diabetes?
- PhysicalActivity: Adults who reported doing physical activity or exercise during the past 30 days other than their regular job.
- GenHealth: Would you say that in general your health is...
- SleepTime: On average, how many hours of sleep do you get in a 24-hour period?
- Asthma: (Ever told) (you had) asthma?
- KidneyDisease: Not including kidney stones, bladder infection or incontinence, were you ever told you had kidney disease?
- SkinCancer: (Ever told) (you had) skin cancer?

## **Summary of data exploration and actions taken for data cleaning and feature engineering**

Data was already preprocessed

```
> data.info()
# The data is already preprocessed
[3] ✓ 0.2s

<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
```

## Label Encoding

```
In [8]: print("Name \t\t Counts \t Values ")
for col in list(data.columns[data.dtypes==object]):
    print(col , len(data[col].unique()) , list(data[col].unique()) )
```

```
Name           Counts           Values
HeartDisease 2 ['No', 'Yes']
Smoking 2 ['Yes', 'No']
AlcoholDrinking 2 ['No', 'Yes']
Stroke 2 ['No', 'Yes']
DiffWalking 2 ['No', 'Yes']
Sex 2 ['Female', 'Male']
Race 6 ['White', 'Black', 'Asian', 'American Indian/Alaskan Native', 'Other', 'Hispanic']
Diabetic 2 ['Yes', 'No']
PhysicalActivity 2 ['Yes', 'No']
GenHealth 5 ['Very good', 'Fair', 'Good', 'Poor', 'Excellent']
Asthma 2 ['Yes', 'No']
KidneyDisease 2 ['No', 'Yes']
SkinCancer 2 ['Yes', 'No']
```

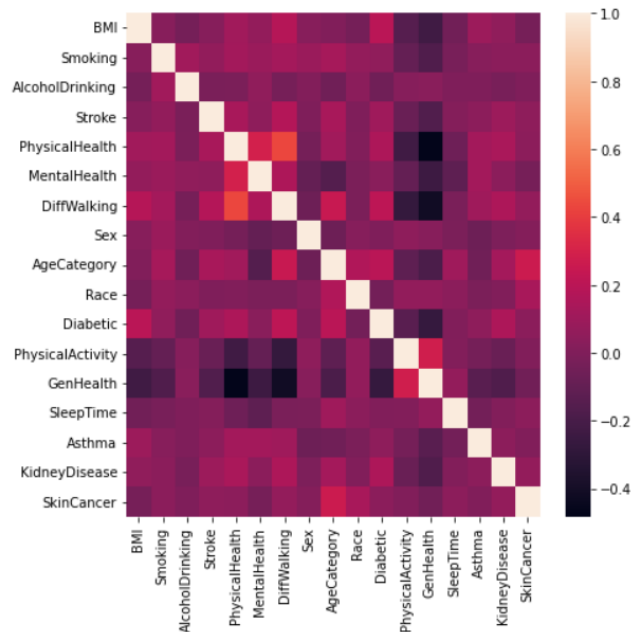
```
In [10]: from sklearn.preprocessing import LabelEncoder
for col in data.columns:
    if data[col].dtype == 'O':
        le = LabelEncoder()
        data[col] = le.fit_transform(data[col])
```

```
data.info()
✓ 0.6s

<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 int32
1   BMI                    319795 non-null float64
2   Smoking                319795 non-null int32
3   AlcoholDrinking        319795 non-null int32
4   Stroke                 319795 non-null int32
5   PhysicalHealth          319795 non-null float64
6   MentalHealth            319795 non-null float64
7   DiffWalking            319795 non-null int32
8   Sex                    319795 non-null int32
9   AgeCategory            319795 non-null int64
10  Race                   319795 non-null int32
11  Diabetic                319795 non-null int32
12  PhysicalActivity        319795 non-null int32
13  GenHealth               319795 non-null int64
14  SleepTime               319795 non-null float64
15  Asthma                  319795 non-null int32
16  KidneyDisease           319795 non-null int32
17  SkinCancer              319795 non-null int32
dtypes: float64(4), int32(12), int64(2)
memory usage: 29.3 MB
```

EDA  
Correlation matrix

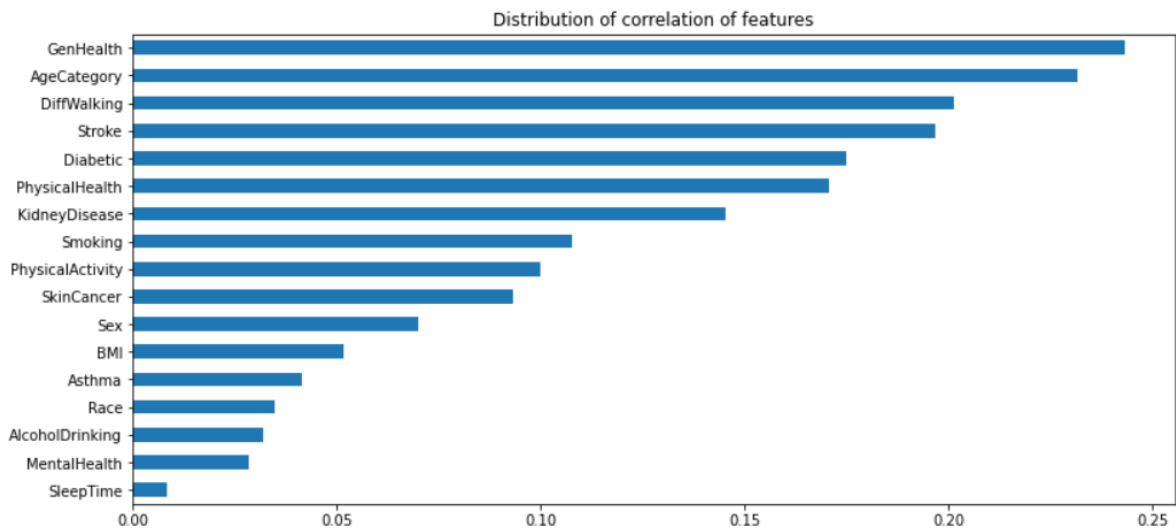
Out[54]: <AxesSubplot:>



*We can notice a higher correlation of 0.5 between physical health and difficulty in walking which kinda makes sense*

```
plt.figure(figsize = (13,6))
plt.title('Distribution of correlation of features')
abs(data.corr()['HeartDisease']).sort_values()[:1].plot.barh()
```

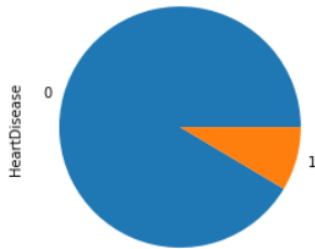
<AxesSubplot:title={'center':'Distribution of correlation of features'}>



### ***Analysis on Target Variable***

```
: ax=data['HeartDisease'].value_counts(normalize=True).plot(kind='pie')
data['HeartDisease'].value_counts(normalize=True)
# We can see that the target variable is skewed and thus we will have to use stratified shuffle split

: 0    0.914405
  1    0.085595
Name: HeartDisease, dtype: float64
```



## **Summary of Models Trained**

### ***Logistic Regression***

```

from sklearn.linear_model import LogisticRegression
lr = LogisticRegression(class_weight={0:1 , 1:20})
lr.fit(X_train,y_train)
y_pred = lr.predict(X_test)

from sklearn.metrics import precision_recall_fscore_support as score
from sklearn.metrics import confusion_matrix, accuracy_score, roc_auc_score

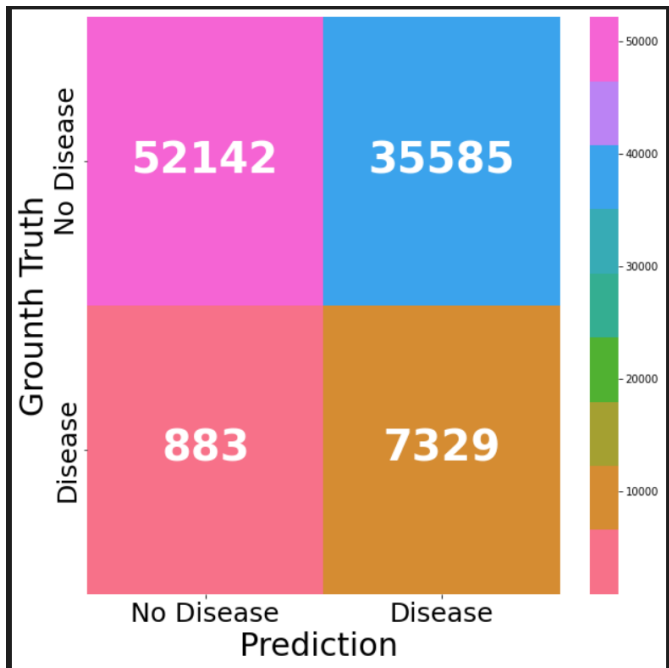
confusion_matrix(y_test,y_pred)

array([[52142, 35585],
       [ 883, 7329]], dtype=int64)

from sklearn.metrics import classification_report, f1_score
print(classification_report(y_test, y_pred))
print('Accuracy score: ', round(accuracy_score(y_test, y_pred), 2))
print('F1 Score: ', round(f1_score(y_test, y_pred), 2))

```

	precision	recall	f1-score	support
0	0.98	0.59	0.74	87727
1	0.17	0.89	0.29	8212
accuracy			0.62	95939
macro avg	0.58	0.74	0.51	95939
weighted avg	0.91	0.62	0.70	95939



## SVM

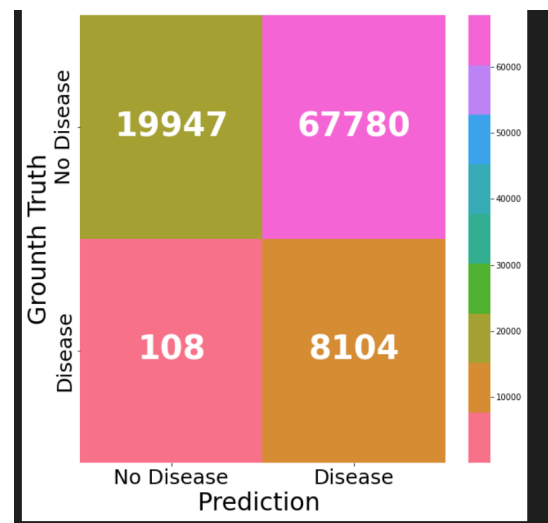


```
## Linear SVM
from sklearn.svm import LinearSVC
svm = LinearSVC(class_weight={0:1, 1:10})
svm.fit(X_train,y_train)
svm_pred = svm.predict(X_test)
```

D:\Computer science\lib\site-packages\sklearn\svm\\_base.py:1206: Convergence warnings.warn(

```
print(classification_report(y_test, svm_pred))
print('Accuracy score: ', round(accuracy_score(y_test, svm_pred), 2))
print('F1 Score: ', round(f1_score(y_test, svm_pred)))
```

	precision	recall	f1-score	support
0	0.99	0.23	0.37	87727
1	0.11	0.99	0.19	8212
accuracy			0.29	95939
macro avg	0.55	0.61	0.28	95939
weighted avg	0.92	0.29	0.35	95939



## Random Forest

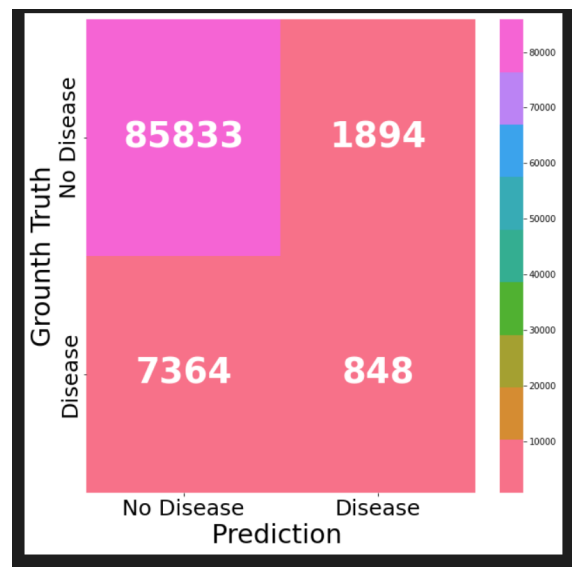
```
from sklearn.ensemble import RandomForestClassifier
RF = RandomForestClassifier(oob_score=True,
                           random_state=42,
                           warm_start=True,
                           n_jobs=-1,
                           class_weight={0:1,1:10} )
RF.fit(X_train,y_train)
```

RandomForestClassifier(class\_weight={0: 1, 1: 10}, n\_jobs=-1, oob\_score=True, random\_state=42, warm\_start=True)

```
rf_pred = RF.predict(X_test)

print(classification_report(y_test, rf_pred))
print('Accuracy score: ', round(accuracy_score(y_test, rf_pred), 2))
print('F1 Score: ', round(f1_score(y_test, rf_pred)))
```

	precision	recall	f1-score	support
0	0.92	0.98	0.95	87727
1	0.31	0.10	0.15	8212
accuracy			0.90	95939
macro avg	0.62	0.54	0.55	95939
weighted avg	0.87	0.90	0.88	95939



### *Summary of above models(Observation)*

- So far, the logistic regression model with high class weights did the best for our needs of classifying whether a patient will have a heart disease or not though it has a high percent of classifying a person has heart disease who might not have heart disease
- The SVM model was better than logistic regression in classifying a person with heart who has heart disease but did extremely worse in False positive cases
- Random Forest did better than KNN on classifying if a person doesn't have heart disease but failed to classify people with heart disease

## **Recommended Model**

I recommend the Logistic Regression model since the need to classify a patient who might have heart disease is more important

## **Key Findings and Insights**

It was really surprising to know that nearly 10 percent of patients will have a myocardial infarction

Skin Cancer and Race has higher significance than sleep time in heart disease

## **Suggestions for next steps**

- > PCA
- > Decision Trees
- > Feature Selection
- > Smote
- > Voting Classifiers