9_Predicting_Heart_Disease_Using_Logistic_Regression_and_ML_

July 6, 2024

0.1 Heart Disease Prediction Using Machine Learning: A Logistic Regression Approach - Vignesh Prabhu

This project focuses on predicting heart disease using machine learning, specifically employing logistic regression. Clinical data was analyzed to identify key risk factors and develop an accurate and reliable predictive model. The goal is to assist healthcare professionals in early diagnosis and effective treatment planning of heart disease.

Import Depedencies

[36]:
```
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.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

Data Collection and Preprocessing

[37]:
```
#Load the data to Pandas dataframe
heart_data=pd.read_csv("/content/heart_disease_data.csv")
```

[38]:
```
# To print first 5 data's in dataset
heart_data.head()
```

[38]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
4   0     2       1
```

[39]:
```
#To check Rows and columns
heart_data.shape
```

[39]:
```
(303, 14)
```

[40]:
```
#To check informations
heart_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        303 non-null    int64
 12  thal      303 non-null    int64
 13  target    303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

[41]:
```
#statistical datas
heart_data.describe()
```

[41]:

	age	sex	cp	trestbps	chol	fbs	\
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	

	restecg	thalach	exang	oldpeak	slope	ca	\
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	

```
mean      0.528053  149.646865    0.326733    1.039604    1.399340    0.729373
std       0.525860   22.905161    0.469794    1.161075    0.616226    1.022606
min       0.000000   71.000000    0.000000    0.000000    0.000000    0.000000
25%       0.000000  133.500000    0.000000    0.000000    1.000000    0.000000
50%       1.000000  153.000000    0.000000    0.800000    1.000000    0.000000
75%       1.000000  166.000000    1.000000    1.600000    2.000000    1.000000
max       2.000000  202.000000    1.000000    6.200000    2.000000    4.000000

             thal      target
count  303.000000  303.000000
mean     2.313531    0.544554
std      0.612277    0.498835
min      0.000000    0.000000
25%      2.000000    0.000000
50%      2.000000    1.000000
75%      3.000000    1.000000
max      3.000000    1.000000
```

[42]:
```
#To check any Null values in Data
heart_data.isnull().sum()
```

[42]:
```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

[43]:
```
#checking Distribution of Target values
heart_data['target'].value_counts() # 1-Defective Heart 0- healthy heart
```

[43]:
```
target
1    165
0    138
Name: count, dtype: int64
```

Spliting Feautres and Target

[44]:
```
X=heart_data.drop(columns='target',axis=0)
Y=heart_data['target']
```

[45]:
```
print(X)
```

```
     age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0     63    1   3       145   233    1        0      150      0      2.3
1     37    1   2       130   250    0        1      187      0      3.5
2     41    0   1       130   204    0        0      172      0      1.4
3     56    1   1       120   236    0        1      178      0      0.8
4     57    0   0       120   354    0        1      163      1      0.6
..   ...  ...  ..       ...   ...  ...      ...      ...    ...      ...
298   57    0   0       140   241    0        1      123      1      0.2
299   45    1   3       110   264    0        1      132      0      1.2
300   68    1   0       144   193    1        1      141      0      3.4
301   57    1   0       130   131    0        1      115      1      1.2
302   57    0   1       130   236    0        0      174      0      0.0

     slope  ca  thal
0        0   0     1
1        0   0     2
2        2   0     2
3        2   0     2
4        2   0     2
..     ...  ..   ...
298      1   0     3
299      1   0     3
300      1   2     3
301      1   1     3
302      1   1     2

[303 rows x 13 columns]
```

[46]:
```
print(Y)
```

```
0      1
1      1
2      1
3      1
4      1
      ..
298    0
299    0
300    0
301    0
302    0
Name: target, Length: 303, dtype: int64
```

Spliting Training and Testing data

[47]:
```
X_train, X_test, Y_train , Y_test=train_test_split(X,Y,test_size=0.
↪2,stratify=Y,random_state=2) #Stratify used for split data even manner
```

[48]:
```
print(X.shape,X_train.shape,X_test.shape)
```

```
(303, 13) (242, 13) (61, 13)
```

Model Training

[49]:
```
logistic_model=LogisticRegression()
```

[50]:
```
logistic_model.fit(X_train,Y_train)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[50]:
```
LogisticRegression()
```

Model Evaluation

[51]:
```
#accuracy score
X_train_prediction=logistic_model.predict(X_train)
training_data_accuracy=accuracy_score(X_train_prediction,Y_train)
print("Accuracy on training data:",training_data_accuracy)
```

```
Accuracy on training data: 0.8512396694214877
```

[52]:
```
#Accuracy For test data
X_test_prediction=logistic_model.predict(X_test)
test_data_accuracy=accuracy_score(X_test_prediction,Y_test)
print("Accuracy on test data:",test_data_accuracy)
```

```
Accuracy on test data: 0.819672131147541
```

Bulding Predictive System

[57]:
```
input_data=(44,1,2,130,233,0,1,179,1,0.4,2,0,2,)
#change input data to numpy array
input_data_as_numpy_array=np.asarray(input_data)
#reshape the numpy array as we are predicting for only one instance
```

```
input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)
prediction=logistic_model.predict(input_data_reshaped)
print(prediction)
if(prediction[0]==0):
  print("The Person does not have a Heart Disease")
else:
  print("The Person has Heart Disease")
```

```
[1]
The Person has Heart Disease

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but LogisticRegression was fitted with feature
names
  warnings.warn(
```

[58]:
```
input_data=(62,0,0,140,268,0,0,160,0,3.6,0,2,2)
#change input data to numpy array
input_data_as_numpy_array=np.asarray(input_data)
#reshape the numpy array as we are predicting for only one instance
input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)
prediction=logistic_model.predict(input_data_reshaped)
print(prediction)
if(prediction[0]==0):
  print("The Person does not have a Heart Disease")
else:
  print("The Person has Heart Disease")
```

```
[0]
The Person does not have a Heart Disease

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but LogisticRegression was fitted with feature
names
  warnings.warn(
```

This project developed a heart disease prediction model using logistic regression, which demonstrated strong predictive accuracy. By analyzing clinical data, we identified key risk factors and created a reliable tool for early diagnosis. Future work could enhance the model by integrating additional algorithms and more diverse datasets, further supporting healthcare professionals in improving patient outcomes.

0.2 Thank You!