Artificial Intelligence & Soft Computing Lab

Heart Disease Prediction using Neural Network

November 12, 2022,

Made by

Harshit Sharma 209301269

“You can have Data without information but you cannot have information without Data”

- Daniel Keys Moran

Introduction

Heart disease is a general term that includes many heart problems. It's also called cardiovascular disease, which means heart and blood vessel disease. The causes of heart

disease depend on the kind of disease. Some possible reasons include lifestyle, genetics, infections, medicines, and other diseases.

Problem Statement

To build a neural network model to predict efficiently whether a person is suffering from heart disease based on the attributes given in the dataset collected from Kaggle.

Approach to Solution

To build a Machine learning model using Google Colaboratory which is an online platform for Machine Learning Development. The language used to build the model is Python3. We would be using technologies like TensorFlow, NumPy, seaborn, Keras, and matplotlib. We would be using Dataset imported from Kaggle which is a free, open-source platform containing a large variety of datasets to choose from.

About the technologies used

1. **TensorFlow:** The TensorFlow platform helps you implement best practices for data automation, model tracking, performance monitoring, and model retraining. Using production-level tools to automate and track model training over the lifetime of a product, service, or business process is critical to success.

2. **Keras:** Keras allows users to productize deep models on smartphones (iOS and Android), on the web, or on the Java Virtual Machine. It also allows the use of distributed training of deep-learning models on clusters of Graphics processing units (GPU) and tensor processing units (TPU).

3. **Seaborn:** Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and color palettes to make statistical plots more attractive. It is built on the top of the Matplotlib library and is also closely integrated into the data structures from pandas.

4. **NumPy & Pandas:** NumPy stands for Numerical Python and it is a core scientific computing library in Python. It provides efficient multi-dimensional array objects and various operations to work with these array objects.

   Pandas is an open-source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays

Code Snippets

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn import preprocessing
```

```
[2]  !pip install utils
```

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

```
[5]  import matplotlib
     #matplotlib.use("TkAgg")
     #from utils import preprocess
     import pandas as pd
     import seaborn as sns
     from sklearn.model_selection import train_test_split
     from sklearn.metrics import confusion_matrix
     from sklearn.metrics import roc_curve
     from keras.models import Sequential
     from keras.layers import Dense,Flatten
     import matplotlib.pyplot as plt
     import numpy as np
     np.random.seed(16)

     df = pd.read_csv('/content/drive/MyDrive/Projects/Project 9 : Heart Disease prediction/heart_disease_data.csv')
```

```
[6]  df.head()
```

```
     df.info()
```

```
[8]  X = df.iloc[:,:13].values
     y = df["target"].values
```

```
[9]  X_train,X_test,y_train, y_test = train_test_split(X,y,test_size = 0.2 , random_state = 0 )
```

```
[10] from sklearn.preprocessing import StandardScaler
     sc = StandardScaler()
     X_train = sc.fit_transform(X_train)
     X_test = sc.transform(X_test)
```

```
[11] classifier = Sequential()
     classifier.add(Dense(activation = "relu", input_dim = 13,
                          units = 8, kernel_initializer = "uniform"))
     classifier.add(Dense(activation = "relu", units = 14,
                          kernel_initializer = "uniform"))
     classifier.add(Dense(activation = "sigmoid", units = 1,
                          kernel_initializer = "uniform"))
     classifier.compile(optimizer = 'adam' , loss = 'binary_crossentropy',
                        metrics = ['accuracy'] )
```

```
[12] classifier.fit(X_train , y_train , batch_size = 8 ,epochs = 100,verbose=False )
```

```
[13] y_pred = classifier.predict(X_test)
     y_pred = (y_pred > 0.5)
```

```
[14] cm = confusion_matrix(y_test,y_pred)
     cm
     ax = sns.heatmap(cm, annot=True, xticklabels=['No Heart Disease', 'Heart Disease'], yticklabels=['No Heart Disease', 'Heart Disease'], cbar=False, cmap='Blues')
     ax.set_xlabel("Prediction")
     ax.set_ylabel("Actual")
     plt.show()
     plt.clf()
```

```
[15] accuracy = (cm[0][0]+cm[1][1])/(cm[0][1] + cm[1][0] +cm[0][0] +cm[1][1])
     print(accuracy*100)
```

```
scores = classifier.evaluate(X_train, y_train, verbose=False)
print("Training Accuracy: %.2f%%\n" % (scores[1]*100))
scores = classifier.evaluate(X_test, y_test, verbose=False)
print("Testing Accuracy: %.2f%%\n" % (scores[1]*100))
```

```
from tensorflow import keras
#from keras import Model
y_test_pred_probs = classifier.predict(X_test)
FPR, TPR, _ = roc_curve(y_test, y_test_pred_probs)
plt.plot(FPR, TPR)
plt.plot([0,1],[0,1],'--', color='black') #diagonal line
plt.title('ROC Curve')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.show()
plt.clf()
```

Output

Training and Testing Accuracy

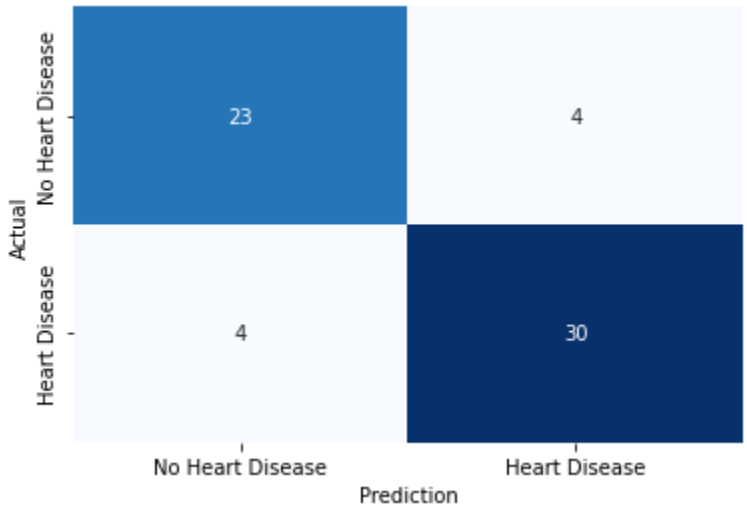
```
Training Accuracy: 89.67%

Testing Accuracy: 86.89%
```

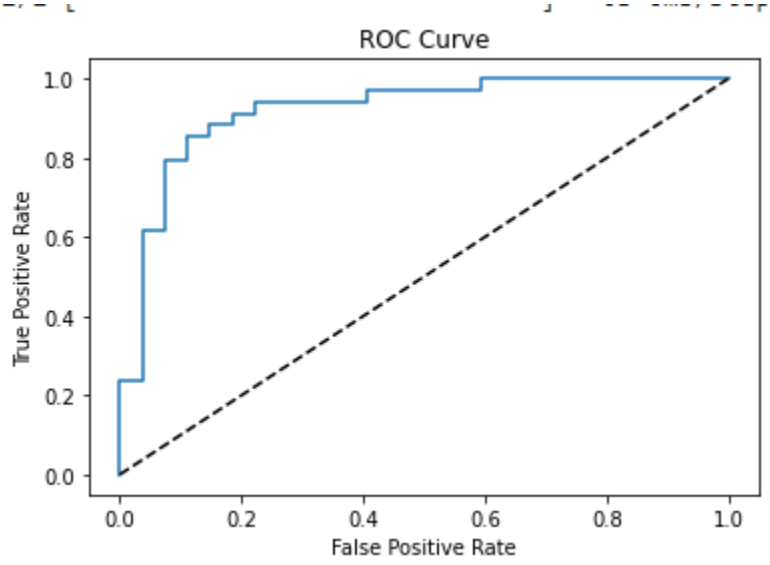
Dataset

	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

Confusion Matrix


23
4
4
30
No Heart Disease
Heart Disease
Actual
No Heart Disease
Heart Disease
Prediction

ROC Curve


ROC Curve
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

Conclusion

The model Neural Network model created using Python3, TensorFlow and Keras predict if the person is suffering from Heart Disease or is at a potential risk of having Heart Disease with an accuracy of 86.2%.

Thus, the model created in this project can be implemented for real-world applications and can be extended further by training a larger dataset and improving the accuracy of prediction using advanced methods of Deep Learning and Convolutional Neural Networks.