**A  REPORT**

*on*

**NAMEOF PROJECT: Heart Disease Prediction**

**SUBMITTED TO:**

Allsoft Solutions Pvt. Ltd.

**SUBMITTED BY**

TUSHAR BHATIA

Session: JUNE 2022

Batch : IBM Artificial Intelligence

**COLLEGE:**

HMR Institute of Technology and Management,

Delhi

**TABLE** **OF** **CONTENTS**

**1.** **Introduction**

**2.** **Software** **Requirements** **and** **Specifications** **(SRS)** **Document**

**3.** **Software** **and** **Hardware** **Requirements**

**4. Planning of work**

**5. System Architecture**

**6.** **Coding**

**7.** **Testing**

**8.Implementation**

**9.** **Snapshots**

**10.** **Bibliography**

**INTRODUCTION**

My project title is “Heart Disease Prediction web application”. I have tried to make the complicated process of Disease prediction as simple as possible using Structured & Modular technique & Selection based web interface. I have tried to design the Project in such a way that user may not have any difficulty in using this package & further expansion is possible without much effort. Though this project is not fully accurate, but the main purpose of it is to predict the chances of heart disease based on a variety of factors.

Machine Learning model is software that can learn and respond through experiences. Machine Learning has a wide range of applications in various fields of technology and science. Many top companies give preference to machine learning as one of the most important domains. For example, Amazon, Netflix, Facebook, YouTube all these companies use Machine Learning algorithms to improve customer experience. Since Machine Learning is a much established and more blooming field, it is a great career option one may choose.  
The goal of the Machine Learning model is to make use of provided data and make better predictions. There are some steps to build a machine learning model. One of the most important and final steps in building a Machine Learning project is Model deployment. There are many frameworks available for deploying the Machine learning model on the web. Some of the most used Python frameworks are Django and Flask. But these frameworks require a little knowledge of languages such as HTML, CSS, and JavaScript. So, a new framework known as Streamlit was introduced to deploy the Machine Learning model without the need of having the knowledge of Front-End Languages. It is quite easy to deploy using Streamlit.

**SOFTWARE REQUIREMENTS AND SPECIFICATIONS (SRS) DOCUMENT**

**PURPOSE**

The purpose of this document is to present a detailed description of Heart Disease Prediction. Machine Learning is very useful in the medical field to detect many diseases in the early stage. The use of predictive analytics is gaining popularity in Disease prediction. In this thesis, we will apply techniques from machine learning to Heart disease dataset to analyze the various factors and predict the final result.

**Document Conventions**

This SRS archive keeps up diverse sorts of fonts, sizes, and faces. This record additionally specified Times New Roman: font sort is Bold and Normal.

**Project Scope**

In this project, we will use the dataset obtained from the popular Kaggle website. The machine learning model uses a dataset of about 350 individual records of real-world patients to calculate this probability

The proposed project consists of a web-based application that would allow individuals to obtain the predicted potential of attaining such CVDs during their lifetime through a machine-learning model.

The proposal consists of supervised machine learning model which individually test the input given by the user to identify any possible likelihood that they could attain such conditions.

**Objectives of the project**

In this thesis, we use a machine learning model web-based interface to predict of the chances of CVDs in individuals based on a variety of factors

**Overall Description**

The machine learning model uses data set that dates from 1988 and consists of four databases: Cleveland, Hungary, Switzerland, and Long Beach V. It contains 76 attributes, including the predicted attribute, but all published experiments refer to using a subset of 14 of them. The "target" field refers to the presence of heart disease in the patient. It is integer valued 0 = no disease and 1 = disease.

The models select 14 attributes such as age, sex, chest pain, resting blood pressure, resting blood sugar, serum cholesterol, etc., and compare that data to whether that individual has attained any form CVD at any point during their lifetime. Different models provide different accuracies based on how they are trained. However, as the number of users rise and each user adds their individual data to the model, the accuracy of the proposed model will ultimately increase and provide users with dependable observations and allow cardiologists to adapt their techniques to better suit the condition of the person being diagnosed.

**PRINCIPLES** **OF** **SYSTEM** **ANALYSIS**

1. Understand the problem before you begin to create the prediction model.

2. Develop prototypes that enable a user to understand how human machine interaction will occur.

3. Record the origin of and the reason for every requirement.

4. Use multiple views of requirements like building data, function and behavioural models.

5. Work to eliminate ambiguity.

**HARDWARE AND SOFTWARE REQUIREMENTS**

Machine learning is a subset of artificial intelligence function that provides the system with the ability to learn from data without being programmed explicitly. Machine learning is basically a mathematical and probabilistic model which requires tons of computations. It is very trivial for humans to do those tasks, but computational machines can perform similar tasks very easily. When trying to gain business value through machine learning, access to best hardware that supports all the complex functions is of utmost importance. With a variety of CPUs, GPUs, TPUs, and ASICs, choosing the right hardware may get a little confusing. This blog discusses hardware consideration when building an infrastructure for machine learning projects. Python leads the pack, with 57% of data scientists and machine learning developers using it and 33% prioritizing it for development.

**Hardware Requirements**

Processor: Intel ® Pentium ®

RAM: 4 GB and above

Hard Disk: 120 GB Speed: 1.6 GHz and above

Monitor: 15’’ LED SVGA

Input Devices: Keyboard, Mouse

**Software Requirements**

Operating System: Windows 8/8.1/10/11

Coding Language: Python

Platform: Chrome v50 and above

Special Tools: Visual Studio Code, Jupyter Notebooks

**PLANNING OF WORK**

Machine learning is a subset of artificial intelligence function that provides the system with the ability to learn from data without being programmed explicitly.

Machine learning is basically a mathematical and probabilistic model which requires tons of computations. It is very trivial for humans to do those tasks, but computational machines can perform similar tasks very easily.

The following are the phases in the modelling process:

1.Examine the problem. We must first thoroughly investigate the situation to pinpoint the problem and fully comprehend its central issues

2.Construct a model

3.Complete the model

4.Verify and interpret the solution of the model

5.Prepare a report on the model

6.Keep the model going

**SYSTEM ARCHITECTURE**

The proposed system consists of a developer who is responsible for developing the machine learning model with the help of the dataset obtained from the hospital. The end user can access the built model from the User Interface which was made by the developer.

The dataset which is collected from the Hospital is stored in the database and is used for building the machine learning models. The model is then trained and then deployed in the web application as the user interface for the end users to interact with.

Diagram

Description automatically generated

**CODING**

**Model Creation:**

#Importing the libraries

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

#Loading the data

data = pd.read\_csv ("heart.csv")

#Print the first 5 rows of the dataset

data.head()

Table

Description automatically generated

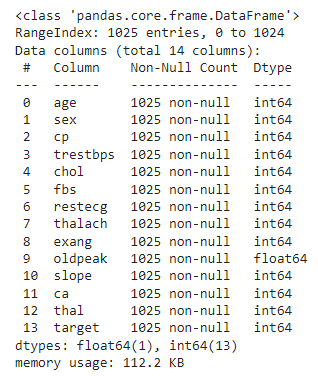
#To get the number of rows and columns in the dataset

data.shape

***Outpu*t:**  
(1025, 14)

#Getting information about the data

data.info()



#Checking the distribution of Target Variable

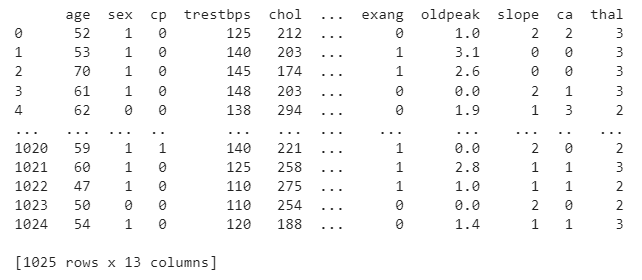
data["target"].value\_counts()

***Outpu*t:**  
1 5260 499Name: target, dtype: int64

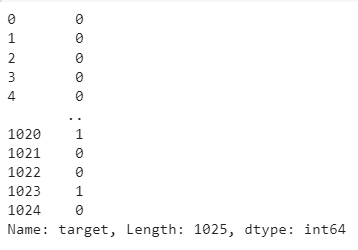
X = data.drop (columns='target', axis=1)

Y = data ['target']

print(X)



Print(Y)



#Splitting the dataset

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split (X, Y, test\_size=0.2, stratify=Y, random\_state=2)

print (X.shape, X\_train.shape, X\_test.shape)

***Output:***

(1025, 13) (820, 13) (205, 13)

#Initializing model object

model = LogisticRegression()

# Training the LogisticRegression model with the Training data

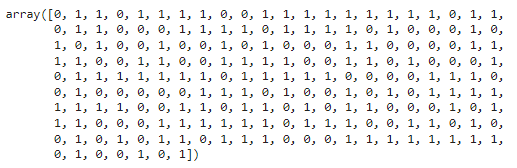
model.fit (X\_train.values, Y\_train)

***Output:*** LogisticRegression()

# Predicting the output with test data

y\_pred=model.predict (X\_test.values)

print (y\_pred)



#Calculating the accuracy of the predicted outcome

print (accuracy\_score (Y\_test,y\_pred))

***Output***: 0.8048780487804879

#Building a predictive system

input\_data = (62, 0, 0, 140, 268, 0, 0, 160, 0, 3.6, 0, 2, 2)

# Change the input data to a numpy array

numpy\_data= np.asarray(input\_data)

# reshape the numpy array as we are predicting for only on instance

input\_reshaped = numpy\_data.reshape (1,-1)

prediction = model.predict (input\_reshaped)

if (prediction[0]== 0):

print ('The Person does not have a Heart Disease')

else:

print ('The Person has Heart Disease')

***OUTPUT:*** The Person does not have a Heart Disease

#Saving the trained model

import pickle

filename = "trained\_model.sav"

#dump=save your trained model

pickle.dump (model,open (filename,"wb"))

#loading the saved model

loaded\_model = pickle.load (open ("trained\_model.sav","rb"))

**Python Code to Deploy Machine Learning model using Streamlit**

**app.py**

#Importing the libraries  
import numpy as np  
import pickle  
import streamlit as st  
  
#loading the saved model  
loaded\_model = pickle.load (open (“C:/Downloads/trained\_model.sav”, ‘rb’))  
  
#Creating a function for Prediction  
def heartdisease\_prediction (input\_data):  
# changing the input data to a numpy array  
numpy\_data= np.asarray (input\_data)  
#Reshaping the numpy array as we are predicting for only on instance  
input\_reshaped = numpy\_data.reshape (1,-1)  
prediction = loaded\_model.predict (input\_reshaped)  
if (prediction[0] == 0):  
st.success (‘The person does not have heart disease’)  
else:  
st.warning (‘The person have heart disease’)  
  
#Adding title to the page  
st.title (‘Heart disease prediction Web App’)  
  
#Getting the input data from the user  
age = st.text\_input (‘Age in Years’)  
sex = st.text\_input (‘Sex : 1 – male, 0 – female’)  
cp = st.text\_input (‘Chest pain type’)  
trestbps = st.text\_input (‘Resting blood pressure in mm Hg’)  
chol = st.text\_input (‘Serum cholesterol in mg/dl’)  
fbs = st.text\_input (‘Fasting blood sugar > 120 mg/dl : 1 – true, 0 – false’)  
restecg = st.text\_input (‘Resting electrocardiographic results’)  
thalach = st.text\_input (‘Maximum heart rate achieved’)  
exang = st.text\_input (‘Exercise induced angina: 1 – yes, 0 – no’)  
oldpeak = st.text\_input (‘ST depression’)  
slope = st.text\_input (‘Slope’)  
ca = st.text\_input (‘Number of major vessels (0-3)’)  
thal = st.text\_input (‘Thal’)  
  
# code for Prediction  
diagnosis = ‘ ‘  
# creating a button for Prediction  
if st.button (‘Heart Disease Test Result’):  
diagnosis=heartdisease\_prediction ([age,sex,cp,trestbps,chol,fbs,restecg,thalach,  
exang,oldpeak,slope,ca,that])

Save the file after pasting the code. And then to deploy using streamlit go to command prompt run the following command.

streamlit run App.py  
(or)  
streamlit run filename.py

**Output:**



**Implementation**

Graphical user interface, text

Description automatically generatedGraphical user interface, application

Description automatically generated

**Screenshots**

Graphical user interface, application

Description automatically generated with medium confidence

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

**Bibliography**

[1] Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and    Techniques to Build Intelligent Systems: Aurélien Géron

[2]  <https://www.kaggle.com/>

[3] https://docs.streamlit.io/