## ARTIFICIAL INTELLIGENCE FINAL PROJECT

On

HEART DISEASE PREDICTION USING MACHINE LEARNING

Submitted By

**SAIKIRAN CHALLA**

## Under the Guidance of

**VAHID BEHZADAN** (Assistant Professor)



# Project Objective:-

According to the CDC, heart disease is one of the leading causes of death for people of most races in the US. 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 indicator includes 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.

Machine learning proves to be effective in assisting in making decisions and predictions from the large quantity of data produced by the health care industry. This project aims to predict future heart disease by analyzing data of patients which classifies whether they have heart disease or not using machine-learning algorithm. Machine Learning techniques can be a boon in this regard. Even though heart disease can occur in different forms, there is a common set of core risk factors that influence whether someone will ultimately be at risk for heart disease or not. By collecting the data from various sources, classifying them under suitable headings & finally analyzing to extract the desired data we can say that this technique can be very well adapted to do the prediction of heart disease.

# APPROACH: -

In our project, we use the following methods

* Naive Bayes

Naive Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems. It is mainly used in text classification that includes a high-dimensional training dataset. Naive Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts based on the probability of an object.

Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles. It is a classification technique based on Bayes’ Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. The Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification method

* Logistic Regression

Logistic Regression is a supervised learning that computes the probabilities for classification problems with two outcomes. It can also be extended to predict several classes. In Logistic Regression model, we apply the sigmoid function, This function successfully maps any number into the value between 0 and 1 and we can regard this value as the probability of predicting classes. For example, we have two classes and they are presence of heart disease and absence of disease. If we set the threshold as 0.5, applying the sigmoid function gives us a value of 0.7, which means the man has the 70% probability of having heart disease so we will predict that he has heart disease

* Decision Tree

Decision Tree is a Supervised learning technique that can be used for both classification and regression problems, but mostly it is preferred for solving classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules, and each leaf node represents the outcome.

In a Decision Tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. It is called a Decision Tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure. In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm. A Decision Tree simply asks a question and based on the answer (Yes/No), it further split the tree into subtrees. The Decision Tree Algorithm belongs to the family of supervised machine learning algorithms. It can be used for both a classification problem as well as for a regression problem.

Working: In a Decision Tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of the root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node. For the next node, the algorithm again compares the attribute value with the other sub-nodes and moves further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

● Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

● Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).

● Step-3: Divide the S into subsets that contains possible values for the best attributes.

● Step-4: Generate the Decision Tree node, which contains the best attribute.

● Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and call the final node as a leaf node.

##### Goals: -

**Create the model:**

We build a model which gives us most accurate results.

**#Importing required libraries**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np

from scipy import stats

%matplotlib inline

sns.set\_style("whitegrid")

plt.style.use("fivethirtyeight")

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

data.head()

# Description of each attribute

1. age - age in years

2. cp - chest pain type

0: Typical angina: chest pain related decrease blood supply to the heart

1: Atypical angina: chest pain not related to heart

2: Non-anginal pain: typically esophageal spasms (non heart related)

3: Asymptomatic: chest pain not showing signs of disease

3. testbps - resting blood pressure (in mm Hg on admission to the hospital) anything above 130-140 is typically cause for concern

4. Chol - serum cholesterol in mg/dl

5. fbs - (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)

6. restecg - resting electrocardiographic results

7. thalach - maximum heart rate achieved

8. exang - exercise induced angina (1 = yes; 0 = no)

9. old peak - ST depression induced by exercise relative to rest looks at stress of heart during excercise unhealthy heart will stress more

10. slope - the slope of the peak exercise ST segment

11. ca - number of major vessels (0-3) colored by fluoroscopy

12. thal - thalium stress result

13. target - have disease or not (1=yes, 0=no) (= the predicted attribute)

Correlation Matrix

# Let's make our correlation matrix a little prettier

corr\_matrix = data.corr()

fig, ax = plt.subplots(figsize=(10, 10))

ax = sns.heatmap(corr\_matrix,

                 annot=True,

                 linewidths=0.5,

                 fmt=".2f",

                 cmap="YlGnBu");

bottom, top = ax.get\_ylim()

ax.set\_ylim(bottom + 0.5, top - 0.5)

Chart, treemap chart

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Appendix:

Python

Python is an interpreted, high-level, general purpose programming language created by Guido Van Rossum and first released in 1991, Python's design philosophy emphasizes code Readability with its notable use of significant White space. Its language constructs and object oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming.

NumPy

NumPy is a library for the python programming language, adding support for large, multi- dimensional arrays and matrices, along with a large collection of high level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim with contributions from several other developers. In 2005, Travis created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open source software and has many contributors.

Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a statemachine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.