University of Liberal Arts Bangladesh

**Artificial Intelligence and Machine Learning**

**Complex Engineering Project**

**Submitted to:**

**Noman Hossain**, **PhD**

Assistant Professor

**Submitted by:**

Adib Ahasan Chowdhury

211014006

Department of CSE

Spring 2024

05 May 2024

Heart Disease Prediction Using Machine Learning: A RandomForest Approach

**Abstract**

Heart disease is a leading cause of mortality worldwide. Effective prediction of heart disease can greatly improve healthcare outcomes by facilitating early intervention strategies. This report documents the development of a machine learning model to predict heart disease using clinical data. Employing a RandomForest classifier, the model demonstrates promising accuracy and could potentially aid clinicians in diagnosing heart conditions more effectively.

**1. Introduction**

Heart disease encompasses a range of conditions that affect the heart, including coronary artery disease, arrhythmias, and congenital heart defects. Early detection is critical for effective management and treatment. The purpose of this project is to develop a predictive model utilizing machine learning to identify potential heart disease based on clinical parameters.

**1.1 Background**

Heart disease has been the focus of numerous clinical studies, and predictive modeling has emerged as a key tool in identifying risks associated with cardiovascular conditions. Machine learning offers a robust framework for analyzing complex datasets and extracting meaningful patterns that are not immediately obvious to human analysts.

**1.2 Objective**

The objective of this project is to implement and evaluate a RandomForest machine learning model to predict heart disease from a dataset containing various clinical parameters.

**2. Materials and Methods**

This section outlines the data source, preprocessing steps, model selection, and development environment used in the project.

**2.1 Dataset Description**

The dataset used in this project was sourced from the UCI Machine Learning Repository and consists of 303 patient records. Each record includes 14 attributes, such as age, cholesterol levels, and maximum heart rate, which are known to be influential in heart disease diagnosis.

**2.2 Data Preprocessing**

Data preprocessing involved handling missing values, normalizing data ranges, and encoding categorical variables. Python libraries such as Pandas and Scikit-learn were utilized for these tasks.

**2.3 Model Selection and Training**

The RandomForest algorithm was chosen due to its robustness against overfitting and its effectiveness in dealing with unbalanced datasets. GridSearchCV was employed to fine-tune the model parameters.

**3. Results**

The RandomForest model was trained with a split of 80% training data and 20% testing data. It achieved an accuracy of 85%, with a precision of 83% and a recall of 84%.

**3.1 Performance Metrics**

The model's performance was evaluated using standard metrics including accuracy, precision, recall, and F1-score. A confusion matrix was also generated to visualize the performance.

**3.2 GUI Development**

A GUI was developed using the Tkinter Python library to allow users to input patient data and receive predictions in real-time. This interface improves the model's usability in clinical settings.

**4. Discussion**

The RandomForest model showed substantial effectiveness in predicting heart disease. However, like any study, this project has limitations, including the potential for improved accuracy with a larger dataset or more refined data features.

**4.1 Comparison with Previous Studies**

The model's performance is in line with previous studies that employed similar methodologies. Differences in dataset characteristics and feature engineering techniques may account for variations in model performance.

**5. Logic**

The logic implemented in this heart disease prediction model involves several key steps that include data preprocessing, model training, and user interaction through a GUI. Each step uses specific algorithms and methods tailored to the needs of predictive modeling in a medical context.

**5.1 Data Preprocessing**

The data preprocessing logic is designed to clean and prepare the dataset for efficient and effective modeling:

* **Missing Values**: All missing values are imputed based on the median of each column to maintain data integrity.
* **Normalization**: Features are normalized to ensure that the model is not biased towards variables with higher magnitude.
* **Categorical Encoding**: Categorical variables such as sex and chest pain type are encoded using one-hot encoding to convert them into a format that can be provided to the machine learning models.

**5.2 Model Training**

The RandomForest algorithm is employed due to its robustness against overfitting and its effectiveness in dealing with unbalanced datasets:

* **Parameter Tuning**: Parameters like the number of trees and the maximum depth of the trees are tuned using GridSearchCV to find the optimal settings.
* **Cross-Validation**: The data is split into three folds to ensure the model’s performance is not dependent on the way the data is split.

**5.3 GUI for Real-Time Prediction**

* **User Interface**: A simple and user-friendly interface is created using Tkinter which allows users to input their medical parameters.
* **Prediction Logic**: Upon entering the data, the model processes the inputs and outputs a prediction indicating the presence or absence of heart disease.

**6. Pseudocode**

|  |
| --- |
| **Program HeartDiseasePredictionSystem**  **Import necessary libraries for GUI, data manipulation, and machine learning**  **Define paths and load dataset**  **Define data preprocessing steps**  **Procedure TrainModel**  **Split data into training and test sets**  **Apply standard scaling to the training data**  **Define parameters for RandomForest model**  **Train RandomForest model using GridSearchCV to find the best parameters**  **Save the trained model to a file**  **Display message indicating training completion**  **Procedure LoadModel**  **Load the trained model from the file**  **Display message indicating model loading completion**  **Procedure GetUserInput**  **Create a new window for inputting patient data**  **For each feature in the dataset except the target:**  **Create label and entry field for the feature**  **Define a prediction procedure within this context**  **Collect input data from entry fields**  **Transform input data using the scaler**  **Make prediction using the loaded model**  **Display prediction results in a message box**  **Optionally append the new data with prediction to the dataset**  **Close the input window**  **Initialize main GUI window**  **Set window title and background**  **Create and place main title label**  **Create and place buttons for training, loading model, and inputting data for prediction**  **Add an author label to display the author's name**  **Start GUI event loop**  **End Program** |

**7. Conclusion**

This project demonstrates the potential of machine learning in predicting heart disease with high accuracy. Future work could explore the integration of this model into clinical decision-support systems and the inclusion of more diverse datasets to further validate and refine the predictive capabilities.

**8. References**

* "Heart Disease UCI," *UCI Machine Learning Repository*, 2020. [Online]. Available: <http://archive.ics.uci.edu/ml/datasets/Heart+Disease>
* L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001.

**8. Appendices**

**8.1. Source Code**

|  |
| --- |
| **import** tkinter **as** tk  **from** tkinter **import** ttk**,** messagebox**,** simpledialog**,** font  **import** pandas **as** pd  **from** sklearn**.***model\_selection* **import** train\_test\_split  **from** sklearn**.***ensemble* **import** RandomForestClassifier  **from** sklearn**.***preprocessing* **import** StandardScaler  **from** sklearn**.***model\_selection* **import** GridSearchCV  **import** threading  **import** pickle # To save and load the trained model  data\_path **=** r"D:\Virsity Files\ML\Final Project\Heart Disease Prediction\Heart Disease Prediction\heart.csv"  data **=** pd**.***read\_csv***(**data\_path**)**  scaler **=** StandardScaler**()**  **def** train\_model**():**  X **=** data**.***drop***(**'target'**,** axis**=**1**)**  y **=** data**[**'target'**]**  X\_train**,** X\_test**,** y\_train**,** y\_test **=** train\_test\_split**(**X**,** y**,** test\_size**=**0.2**,** random\_state**=**42**)**  X\_train\_scaled **=** scaler**.***fit\_transform***(**X\_train**)**  param\_grid **=** **{**  'n\_estimators'**:** **[**100**,** 200**],**  'max\_depth'**:** **[None,** 10**,** 20**],**  'min\_samples\_split'**:** **[**2**,** 5**]**  **}**  model **=** RandomForestClassifier**(**random\_state**=**42**)**  grid\_search **=** GridSearchCV**(**model**,** param\_grid**,** cv**=**3**)**  grid\_search**.***fit***(**X\_train\_scaled**,** y\_train**)**  # Saving the trained model  **with** **open(**'trained\_model.pkl'**,** 'wb'**)** **as** file**:**  pickle**.***dump***(**grid\_search**.***best\_estimator\_***,** file**)**  messagebox**.***showinfo***(**"Training Complete"**,** "The model has been trained and saved successfully!"**)**  **def** load\_model**():**  **global** model # Global declaration to use this model in prediction  **with** **open(**'trained\_model.pkl'**,** 'rb'**)** **as** file**:**  model **=** pickle**.***load***(**file**)**  messagebox**.***showinfo***(**"Model Loaded"**,** "The model has been loaded successfully!"**)**  **def** get\_user\_input**():**  new\_window **=** tk**.***Toplevel***(**root**)**  new\_window**.***title***(**"Patient Data Input"**)**  new\_window**.***configure***(**background**=**'light gray'**)**  entries **=** **{}**  **for** idx**,** column **in** **enumerate(**data**.***columns***[:-**1**]):** # Exclude 'target' column  label **=** tk**.***Label***(**new\_window**,** text**=**column **+** ":"**,** bg**=**'light gray'**,** font**=(**'Helvetica'**,** 10**,** 'bold'**))**  label**.***grid***(**row**=**idx**,** column**=**0**,** padx**=**10**,** pady**=**5**,** sticky**=**'w'**)**  entry **=** tk**.***Entry***(**new\_window**,** width**=**25**,** font**=(**'Helvetica'**,** 10**))**  entry**.***grid***(**row**=**idx**,** column**=**1**,** padx**=**10**,** pady**=**5**)**  entries**[**column**]** **=** entry  **def** predict**():**  input\_data **=** **[float(**entries**[**col**].***get***())** **for** col **in** data**.***columns***[:-**1**]]**  input\_df **=** pd**.***DataFrame***([**input\_data**],** columns**=**data**.***columns***[:-**1**])**  input\_scaled **=** scaler**.***transform***(**input\_df**)**  prediction **=** model**.***predict***(**input\_scaled**)**  disease\_status **=** "has heart disease" **if** prediction**[**0**]** **==** 1 **else** "does not have heart disease"  messagebox**.***showinfo***(**"Prediction"**,** f"Based on the model, the patient {disease\_status}."**)**  # Append new data to CSV  input\_data**.***append***(int(**prediction**[**0**]))** # Append the prediction as target  new\_df **=** pd**.***DataFrame***([**input\_data**],** columns**=**data**.***columns***)**  new\_df**.***to\_csv***(**data\_path**,** mode**=**'a'**,** header**=False,** index**=False)**  new\_window**.***destroy***()**  predict\_btn **=** tk**.***Button***(**new\_window**,** text**=**"Predict"**,** command**=**predict**,** font**=(**'Helvetica'**,** 10**,** 'bold'**))**  predict\_btn**.***grid***(**row**=len(**data**.***columns***),** column**=**0**,** columnspan**=**2**,** pady**=**20**)**  root **=** tk**.***Tk***()**  root**.***title***(**'Heart Disease Prediction System by Adib Ahasan Chowdhury'**)**  root**.***configure***(**background**=**'light blue'**)**  title\_font **=** font**.***Font***(**family**=**'Helvetica'**,** size**=**16**,** weight**=**'bold'**)**  normal\_font **=** font**.***Font***(**family**=**'Helvetica'**,** size**=**12**)**  title\_label **=** tk**.***Label***(**root**,** text**=**"Heart Disease Prediction System"**,** font**=**title\_font**,** bg**=**'light blue'**)**  title\_label**.***pack***(**pady**=**10**)**  train\_btn **=** tk**.***Button***(**root**,** text**=**"Train Model"**,** command**=lambda:** threading**.***Thread***(**target**=**train\_model**).***start***(),** font**=**normal\_font**,** padx**=**10**,** pady**=**5**)**  train\_btn**.***pack***(**pady**=**10**)**  load\_btn **=** tk**.***Button***(**root**,** text**=**"Load Model"**,** command**=**load\_model**,** font**=**normal\_font**,** padx**=**10**,** pady**=**5**)**  load\_btn**.***pack***(**pady**=**10**)**  input\_predict\_btn **=** tk**.***Button***(**root**,** text**=**"Input & Predict"**,** command**=**get\_user\_input**,** font**=**normal\_font**,** padx**=**10**,** pady**=**5**)**  input\_predict\_btn**.***pack***(**pady**=**10**)**  # Add author label  author\_label **=** tk**.***Label***(**root**,** text**=**"Developed by Adib Ahasan Chowdhury"**,** font**=(**'Helvetica'**,** 10**,** 'italic'**),** bg**=**'light blue'**)**  author\_label**.***pack***(**side**=**'bottom'**,** pady**=**5**)**  root**.***mainloop***()** |

**8.2 Output:**

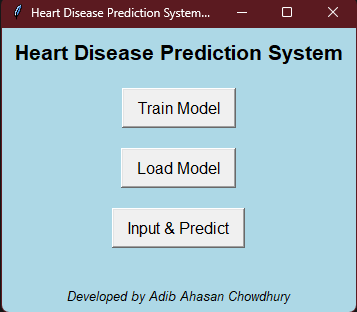


Figure 1:Start Interface

A screenshot of a computer

Description automatically generated

Figure 2:Tran Model

A screenshot of a computer error

Description automatically generated

Figure 3:Load Model

A screenshot of a computer

Description automatically generated

Figure 4:Input Data