Project-1

Heart Disease Prediction Using Machine Learning

Prepared by: Dushyant Kumar

Role: Machine Learning Intern

Abstract

The project aims to develop a model utilizing patient data and Machine Learning algorithms to predict heart disease risk factors and enhance early detection methods. Using a dataset of patient records, we explored the data, built models, and evaluated their performance to identify the best approach.

Introduction

Heart disease leads to significant mortality rates and poses considerable healthcare expenditures globally. Early detection is crucial in mitigating severe cases and reducing healthcare costs, highlighting the urgency for advanced predictive models. Utilizing Machine Learning enables the analysis of extensive datasets to identify high-risk individuals cost-effectively and at scale, revolutionizing disease prediction.

Objective

The primary goal is to construct a dependable model for heart disease prediction based on medical data using machine learning.

Dataset Description

The dataset contains patient demographics, medical history, and test results. It includes the following features:

- Age, Sex
- Chest pain type
- Resting blood pressure, Cholesterol
- Fasting blood sugar
- Resting ECG results, Maximum heart rate achieved
- Exercise-induced angina

- ST slope, Oldpeak (ST depression)
- Target (0 = No heart disease, 1 = Heart disease)

Methodology

1. Data Collection: Obtain patient data from diverse sources.
2. Data Preprocessing: Clean and organize the data.
3. Feature Selection: Determine crucial data points.
4. Model Training: Employ Machine Learning algorithms.
5. Model Evaluation: Assess model performance.
6. Deployment: Implement in healthcare settings.
7. Prediction: Tested the model with new patient data.

Results

1. Logistic Regression Accuracy: 86.89%.
2. Tuned Random Forest Accuracy: 94.96%.
   - Best Parameters: {'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 300}.
   - Precision and Recall were excellent for both classes.
3. Feature Importance: The key predictors included 'ST slope', 'age', and 'cholesterol'.

Discussion

The results demonstrate that the Random Forest model performs exceptionally well in predicting heart disease, with an accuracy of 94.96%. Key features such as ST slope, cholesterol levels, and age were identified as the most influential predictors. Despite the high accuracy, the model's effectiveness is limited to the quality and diversity of the dataset. Further testing on unseen data is recommended.

Conclusion

The project successfully demonstrated the application of machine learning in predicting heart disease, offering a promising tool for early diagnosis. The model achieved high accuracy and performance metrics by analyzing various health parameters, showing its potential in real world healthcare settings.

References

1. Dataset source: [Provide dataset reference]
2. Scikit-learn documentation: https://scikit-learn.org/
3. Relevant research papers or articles (if any).

Code Used in the Project

```
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.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.preprocessing import OneHotEncoder
df = pd.read_csv(r"C:\Users\dushy\Downloads\heart csv data.csv")
print(df.head())

# Check for missing values
print(df.isnull().sum())

# Statistical summary of the dataset
print(df.describe())

# Visualize the correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()

# Convert 'sex' column to numerical (0: female, 1: male)
df['sex'] = df['sex'].replace({'male': 1, 'female': 0})

# Convert 'cp' (chest pain type) column to one-hot encoding
```

```
df = pd.get_dummies(df, columns=['cp'], drop_first=True)

# Define features and target variable
X = df.drop('target', axis=1)
y = df['target']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

model = LogisticRegression()
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

# Confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt='d')
plt.title('Confusion Matrix')
plt.show()
```

```
# Classification report

print(classification_report(y_test, y_pred))
```