Transforming Health Care with AI-Powered Disease Prediction

# 1. Overview

This project leverages AI and machine learning to predict the risk of heart disease using patient data from the Cleveland Heart Disease dataset. The Python code below outlines the complete workflow from data preprocessing to model evaluation using Logistic Regression and Random Forest.

# 2. Data Preprocessing

import pandas as pd  
from sklearn.preprocessing import MinMaxScaler  
from sklearn.impute import SimpleImputer  
  
# Load dataset  
df = pd.read\_csv('heart.csv')  
  
# Handle missing values  
imputer = SimpleImputer(strategy='mean')  
df[df.columns] = imputer.fit\_transform(df)  
  
# Normalize numeric features  
scaler = MinMaxScaler()  
df[df.columns] = scaler.fit\_transform(df)  
  
# Encode categorical variables  
df = pd.get\_dummies(df, drop\_first=True)

# 3. Exploratory Data Analysis (EDA)

import seaborn as sns  
import matplotlib.pyplot as plt  
  
# Univariate Analysis  
sns.histplot(df['age'])  
plt.title('Age Distribution')  
  
# Correlation Heatmap  
sns.heatmap(df.corr(), annot=True, fmt='.2f', cmap='coolwarm')  
plt.title('Feature Correlation')  
plt.show()

# 4. Feature Engineering

# Example: Create new feature cholesterol-to-age ratio  
df['chol\_age\_ratio'] = df['chol'] / df['age']  
  
# Binning age into categories  
df['age\_group'] = pd.cut(df['age'], bins=[0, 40, 55, 70, 100], labels=['<40', '40-55', '55-70', '70+'])  
df = pd.get\_dummies(df, columns=['age\_group'], drop\_first=True)

# 5. Model Building and Evaluation

from sklearn.model\_selection import train\_test\_split  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.linear\_model import LogisticRegression  
from sklearn.metrics import classification\_report, confusion\_matrix, roc\_auc\_score  
  
# Split data  
X = df.drop('target', axis=1)  
y = df['target']  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, stratify=y, random\_state=42)  
  
# Train models  
lr = LogisticRegression(max\_iter=1000)  
rf = RandomForestClassifier(n\_estimators=100, random\_state=42)  
  
lr.fit(X\_train, y\_train)  
rf.fit(X\_train, y\_train)  
  
# Evaluate models  
y\_pred\_lr = lr.predict(X\_test)  
y\_pred\_rf = rf.predict(X\_test)  
  
print("Logistic Regression Report:\n", classification\_report(y\_test, y\_pred\_lr))  
print("Random Forest Report:\n", classification\_report(y\_test, y\_pred\_rf))  
print("ROC AUC - RF:", roc\_auc\_score(y\_test, rf.predict\_proba(X\_test)[:,1]))