**Medical Data Analysis (Disease Insights)**

**Objective:** *Predict diseases or gain insights into healthcare trends using data analytics and machine learning.*

**📌 Step 1: Define the Problem Statement**

Before starting, decide on the specific medical problem you want to analyze. Some examples:

* **Disease Prediction** (Diabetes, Heart Disease, Cancer, etc.)
* **Hospital Readmission Analysis**
* **Patient Risk Factor Identification**

For this guide, we will focus on **Heart Disease Prediction**.

**📌 Step 2: Collect a Dataset**

You need a dataset with patient health indicators. Good sources:

* [**Heart Disease Dataset (Kaggle)**](https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset)

💡 **Dataset Features (Example)**:

| **Feature** | **Description** |
| --- | --- |
| Age | Patient's age |
| Sex | Male/Female |
| Blood Pressure | Systolic & Diastolic BP |
| Cholesterol Level | Normal, High, etc. |
| Chest Pain Type | 4 categories |
| Fasting Blood Sugar | > 120 mg/dl (1 = True, 0 = False) |
| Max Heart Rate | Measured during exercise |
| Target (Disease) | 1 = Disease, 0 = No Disease |

**📌 Step 3: Data Cleaning & Preprocessing**

**🔹 Load Data in Python**

import pandas as pd

# Load the dataset

df = pd.read\_csv('heart\_disease.csv')

# Check basic information

df.info()

df.head()

**🔹 Handle Missing Values**

# Check for missing values

print(df.isnull().sum())

# Fill missing values (example: replacing NaN with median)

df.fillna(df.median(), inplace=True)

**📌 Step 4: Exploratory Data Analysis (EDA)**

**🔹 Summary Statistics**

print(df.describe())

**🔹 Data Visualization**

import matplotlib.pyplot as plt

import seaborn as sns

# Correlation Heatmap

plt.figure(figsize=(10,6))

sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt=".2f")

plt.title('Feature Correlation Heatmap')

plt.show()

**🔹 Target Distribution**

sns.countplot(x=df['Target'])

plt.title('Heart Disease Distribution')

plt.show()

**📌 Step 5: Build a Machine Learning Model**

**🔹 Split Data into Train & Test Sets**

python

CopyEdit

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

X = df.drop(columns=['Target']) # Features

y = df['Target'] # Target variable

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

**🔹 Train a Classification Model**

python

CopyEdit

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, classification\_report

# Initialize model

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

# Train model

model.fit(X\_train, y\_train)

# Make predictions

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

# Evaluate performance

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

**📌 Step 6: Model Optimization & Feature Importance**

**🔹 Hyperparameter Tuning**

Use **GridSearchCV** to find the best parameters.

python

CopyEdit

from sklearn.model\_selection import GridSearchCV

param\_grid = {'n\_estimators': [50, 100, 200], 'max\_depth': [None, 10, 20]}

grid\_search = GridSearchCV(RandomForestClassifier(), param\_grid, cv=5)

grid\_search.fit(X\_train, y\_train)

print("Best Parameters:", grid\_search.best\_params\_)

**🔹 Feature Importance**

python

CopyEdit

importances = model.feature\_importances\_

feature\_names = X.columns

plt.figure(figsize=(10,6))

sns.barplot(x=importances, y=feature\_names)

plt.title("Feature Importance")

plt.show()

**📌 Step 7: Data Visualization & Dashboard**

If you want to present insights in **Power BI or Tableau**, export the cleaned data.

df.to\_csv('cleaned\_heart\_data.csv', index=False)

You can create:

* **Bar Charts**: Risk factors by age group
* **Heatmaps**: Correlation of symptoms with disease
* **Pie Charts**: Male vs Female distribution of heart disease

**📌 Step 8: Summary of Requirements**

**📂 Tools Required:**

✅ Python (pandas, sklearn, seaborn, matplotlib)  
✅ Power BI / Tableau (for dashboards)  
✅ Flask / Streamlit (if deployment is needed)