Name: Saniya Deshpande

Email: [saniyadeshpande05@gmail.com](mailto:saniyadeshpande05@gmail.com)

Project: Heart analysis

Without dashboard:

import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
file\_path="Heart Disease data/Heart Disease data.csv"  
df=pd.read\_csv(file\_path)  
  
*# print(df)*print(df.describe())  
  
*# #vizualising the distribution of target variable*sns.countplot(x='target',data=df)  
plt.title("distribution of target variable")  
plt.show()  
  
*#explore distribution of numerical features*column\_names=df.columns.tolist()  
print(column\_names)  
  
numerical\_columns=df.select\_dtypes(include=['int64','float64']).columns.tolist()  
print(numerical\_columns)  
  
num\_features = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']  
df[num\_features].hist(bins=20, figsize=(15, 10))  
plt.suptitle('Distribution of Numerical Features')  
plt.show()  
  
  
*# Scatterplots for individual features against the target*num\_features = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']  
  
for feature in num\_features:  
 plt.figure(figsize=(8, 6))  
 sns.scatterplot(x=feature, y='target', data=df, hue='target', marker='o', s=50)  
 plt.title(f'Scatterplot of {feature} against Target')  
 plt.show()  
  
*# Using box plots for categorical features*cat\_features = ['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal']  
for feature in cat\_features:  
 sns.countplot(x=feature, hue='target', data=df)  
 plt.title(f'Distribution of {feature} colored by Target')  
 plt.show()  
  
*# Correlation heatmap*correlation\_matrix = df.corr()  
plt.figure(figsize=(12, 8))  
sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt='.2f')  
plt.title('Correlation Heatmap')  
plt.show()

With dashboard:  
import dash  
from dash import dcc, html  
from dash.dependencies import Input, Output  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
  
*# Load the dataset*file\_path = "Heart Disease data/Heart Disease data.csv"  
df = pd.read\_csv(file\_path)  
  
*# Initialize the Dash app*app = dash.Dash(\_\_name\_\_)  
  
*# Define the layout of the dashboard*app.layout = html.Div([  
 html.H1("Heart Disease Dataset Dashboard"),  
  
 *# Basic statistics* html.Div([  
 html.H2("Basic Statistics"),  
 html.Pre(id='basic-stats', children=df.describe().to\_string())  
 ]),  
  
 *# Distribution of target variable* html.Div([  
 html.H2("Distribution of Target Variable"),  
 dcc.Graph(  
 id='target-distribution',  
 figure={  
 'data': [  
 {'x': df['target'].value\_counts().index, 'y': df['target'].value\_counts().values, 'type': 'bar'}  
 ],  
 'layout': {  
 'title': 'Distribution of Target Variable'  
 }  
 }  
 )  
 ]),  
  
 *# Histogram of numerical features* html.Div([  
 html.H2("Distribution of Numerical Features"),  
 dcc.Graph(  
 id='numerical-features-histogram',  
 figure={  
 'data': [  
 {'x': df[column], 'type': 'histogram', 'name': column} for column in ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']  
 ],  
 'layout': {  
 'title': 'Distribution of Numerical Features'  
 }  
 }  
 )  
 ]),  
  
 *# Scatterplots for individual features against the target* html.Div([  
 html.H2("Scatterplots of Numerical Features against Target"),  
 dcc.Dropdown(  
 id='scatter-feature-dropdown',  
 options=[{'label': col, 'value': col} for col in ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']],  
 value='age'  
 ),  
 dcc.Graph(id='scatterplot')  
 ]),  
])  
  
*# Define callback for updating scatterplot based on dropdown selection*@app.callback(  
 Output('scatterplot', 'figure'),  
 [Input('scatter-feature-dropdown', 'value')]  
)  
def update\_scatterplot(selected\_feature):  
 return {  
 'data': [  
 {  
 'x': df[selected\_feature],  
 'y': df['target'],  
 'type': 'scatter',  
 'mode': 'markers',  
 'marker': {'size': 8},  
 }  
 ],  
 'layout': {  
 'title': f'Scatterplot of {selected\_feature} against Target',  
 'xaxis': {'title': selected\_feature},  
 'yaxis': {'title': 'Target'}  
 }  
 }  
  
*# Run the app*if \_\_name\_\_ == '\_\_main\_\_':  
 app.run\_server(debug=True)