Summary Report

This is a summary report for the Heart Attack Analysis & Prediction Dataset.

Dataset Description

- Age : Age of the patient
- Sex : Sex of the patient
- exang: exercise induced angina (1 = yes; 0 = no)
- ca: number of major vessels (0-3)
- cp : Chest Pain type chest pain type
    - Value 1: typical angina
    - Value 2: atypical angina
    - Value 3: non-anginal pain
    - Value 4: asymptomatic
- trtbps : resting blood pressure (in mm Hg)
- chol : cholestoral in mg/dl fetched via BMI sensor
- fbs : (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
- rest_ecg : resting electrocardiographic results
    - Value 0: normal
    - Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
    - Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
- thalach : maximum heart rate achieved
- target : 0= less chance of heart attack 1= more chance of heart attack

Data Manipulation Overview

1. Data Ingestion:

- Libraries `pandas` and `matplotlib.pyplot` are imported for data manipulation and visualization.
- A CSV file from a URL is loaded into a DataFrame named `heart`.
- This data is saved locally as "heart.csv".
- View the first few rows, last few rows, column names, and the shape (number of rows and columns) of the DataFrame.

2. Exploratory Data Analysis (EDA):

a. General Analysis:

- Summary statistics (like count, mean, standard deviation, min, quartiles, and max) for the `heart` dataset are generated using the `describe()` method.
- The median value for each column is calculated.

b. Data Filtering:

- A new dataset, `filtered_heart`, is created by filtering out patients whose age is greater than 50.
- The shape, first few rows, and summary statistics of the `filtered_heart` dataset are displayed.

c. Visualization:

- Histograms:
    - A function named `histogram` is defined to generate histograms for each numeric column in the dataset.
    - This function reads the data from the provided file path, iterates over each column, and for each numeric column, it plots a histogram.
    - The function is called for "heart.csv" to display the histograms.
- Scatter Plot:
    - A function named `scatter_age_blood_pressure` is defined to generate a scatter plot between Age and Resting Blood Pressure from the dataset.
    - The function reads the data from the provided file path, extracts Age (1st column) and Resting Blood Pressure (4th column), and plots them against each other.
    - The function is called for "heart.csv" to display the scatter plot.

Ingest

In [ ]:
```python
import pandas as pd
import matplotlib.pyplot as plt
```

In [ ]:
```python
# URL for the CSV file
url = "https://raw.githubusercontent.com/nogibjj/tinayiluo_mini9/main/heart.csv"

# Read the CSV from the URL
heart = pd.read_csv(url)

heart.to_csv("heart.csv", index=False)

# To view the first few rows of the DataFrame
heart.head()
```

Out[ ]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

In [ ]:
```python
# To view the last few rows of the DataFrame
heart.tail()
```

Out[ ]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

In [ ]:
```python
# To view the column names of the DataFrame
heart.columns
```

Out[ ]:
```
Index(['age', 'sex', 'cp', 'trtbps', 'chol', 'fbs', 'restecg', 'thalachh',
       'exng', 'oldpeak', 'slp', 'caa', 'thall', 'output'],
      dtype='object')
```

In [ ]:
```python
# To view the shape of the DataFrame
heart.shape
```

Out[ ]:
```
(303, 14)
```

EDA

Generates Summary Statistics for heart.csv

In [ ]:
```python
heart.describe()
```

Out[ ]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	2.313531	0.544554
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	0.612277	0.498835
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

Calculate the Median Value for Each Column in heart.csv

In [ ]:
```python
heart.median()
```

Out[ ]:
```
age          55.0
sex           1.0
cp            1.0
trtbps      130.0
chol        240.0
fbs           0.0
restecg       1.0
thalachh    153.0
exng          0.0
oldpeak       0.8
slp           1.0
caa           0.0
thall         2.0
output        1.0
dtype: float64
```

Filter the Data To get patients whose age is greater than 50

In [ ]:
```python
# Filter the Data to get patients with age > 50
filtered_heart = heart[heart["age"] > 50]

# Basic EDA on the filtered dataset
print("Shape of the filtered dataset:", filtered_heart.shape)
print("\nFirst 5 rows of the filtered dataset:")
filtered_heart.head()
```

```
Shape of the filtered dataset: (208, 14)

First 5 rows of the filtered dataset:
```

Out[ ]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1

In [ ]:
```python
print("\nDescriptive statistics of the filtered dataset:")
filtered_heart.describe()
```

```
Descriptive statistics of the filtered dataset:
```

Out[ ]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
count	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000	208.000000
mean	59.375000	0.668269	0.918269	134.802885	252.846154	0.187500	0.490385	144.552885	0.375000	1.210096	1.341346	0.889423	2.346154	0.475962
std	5.603118	0.471971	1.062334	18.541104	54.382572	0.391254	0.538296	22.511985	0.485291	1.204558	0.616850	1.017852	0.655990	0.500627
min	51.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	55.000000	0.000000	0.000000	121.500000	215.500000	0.000000	0.000000	130.000000	0.000000	0.100000	1.000000	0.000000	2.000000	0.000000
50%	58.500000	1.000000	0.000000	132.000000	246.000000	0.000000	0.000000	147.500000	0.000000	1.000000	1.000000	1.000000	2.000000	0.000000
75%	63.000000	1.000000	2.000000	145.000000	283.000000	0.000000	1.000000	161.000000	1.000000	1.825000	2.000000	2.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	195.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

Generate Histogram for Each Column in heart.csv

In [ ]:
```python
def histogram(file_path):
    df = pd.read_csv(file_path)
    columns = df.columns

    for column in columns:
        if pd.api.types.is_numeric_dtype(df[column]):  # Check if the column is numeric
            plt.figure(figsize=(8, 4))  # Create a new figure for each column
            plt.hist(
                df[column], bins=30, edgecolor="black", alpha=0.7
            )  # bins parameter added for better visualization
            plt.xlabel(column)
            plt.ylabel("Frequency")
            plt.title(f"Histogram of {column}")
            plt.grid(
                True, which="both", linestyle="--", linewidth=0.5
            )  # Improved grid line appearance
            plt.tight_layout()  # Adjusts subplot params for better layout
            plt.show()  # Display the histogram for the current column


# Call the function
histogram("heart.csv")
```

Generate Scatter Plot For Resting Blood Pressure and Age in heart.csv

In [ ]:
```python
def scatter_age_blood_pressure(file_path):
    df = pd.read_csv(file_path)
    x = df.iloc[:, 0]  # 1st column (age)
    y = df.iloc[:, 3]  # 4th column (resting blood pressure)
    plt.scatter(x, y, alpha=0.5, label="Data Points")
    plt.xlabel("Age")
    plt.ylabel("Resting Blood Pressure (mm Hg)")
    plt.title("Scatter Plot: Age vs. Resting Blood Pressure")
    plt.grid(True)  # Add grid lines for reference
    # plt.legend()
    plt.show()


scatter_age_blood_pressure("heart.csv")
```