

```
In [1]: # Required Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [8]: df = pd.read_csv('heart_disease_uci.csv') # Adjust path if needed
df.columns = [col.lower().strip() for col in df.columns]
```

```
In [9]: df.head()
```

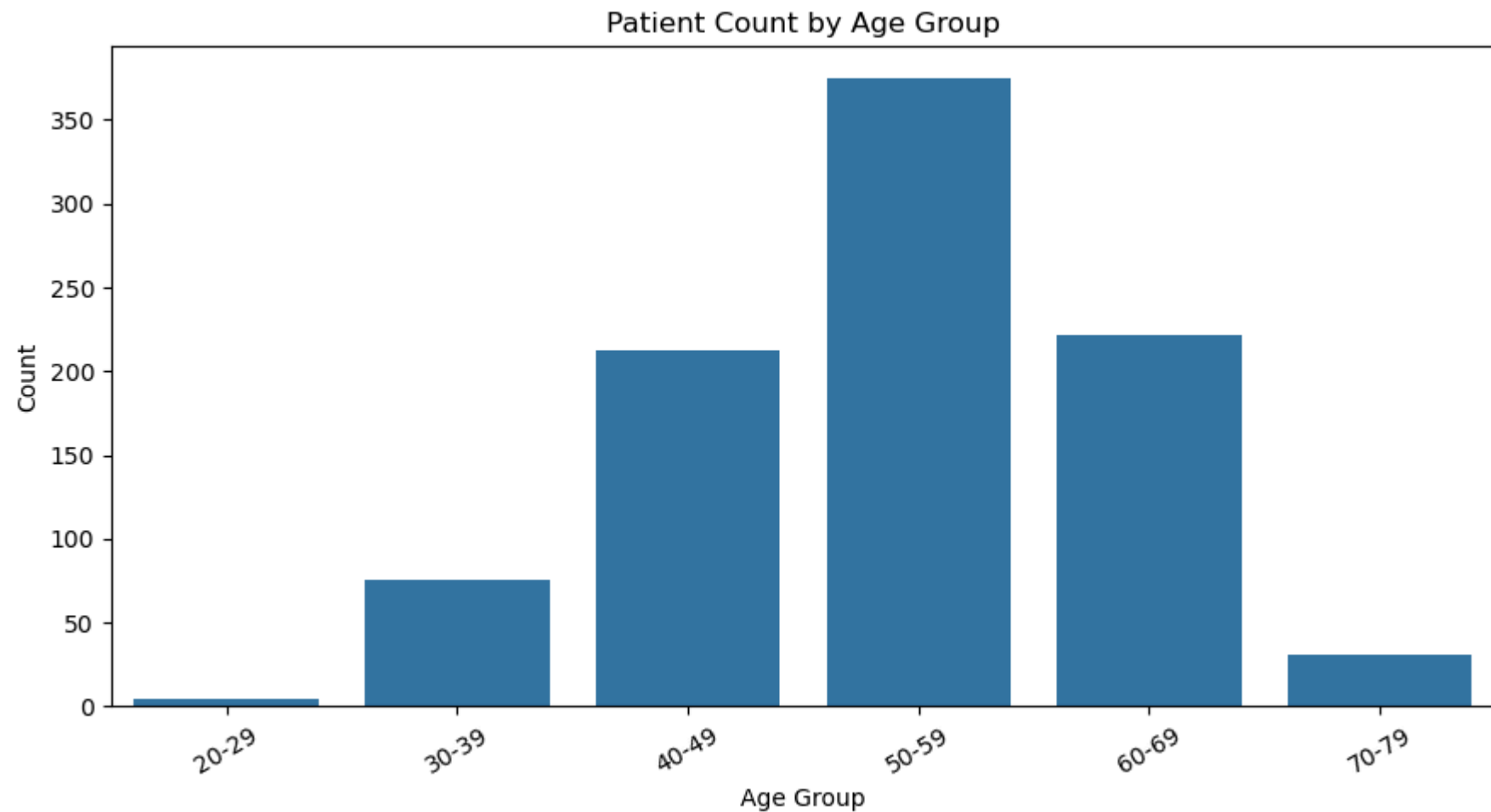
```
Out[9]:
```

	id	age	sex	dataset	cp	trestbps	chol	fbs	restecg	thalch	exang	oldpeak	slope	ca	th
0	1	63	Male	Cleveland	typical angina	145.0	233.0	True	lv hypertrophy	150.0	False	2.3	downsloping	0.0	fixe
1	2	67	Male	Cleveland	asymptomatic	160.0	286.0	False	lv hypertrophy	108.0	True	1.5	flat	3.0	norm
2	3	67	Male	Cleveland	asymptomatic	120.0	229.0	False	lv hypertrophy	129.0	True	2.6	flat	2.0	reversak
3	4	37	Male	Cleveland	non-anginal	130.0	250.0	False	normal	187.0	False	3.5	downsloping	0.0	norm
4	5	41	Female	Cleveland	atypical angina	130.0	204.0	False	lv hypertrophy	172.0	False	1.4	upsloping	0.0	norm

```
In [10]: # Convert boolean strings to actual boolean type if needed
df['fbs'] = df['fbs'].map({'True': 1, 'False': 0})
df['exang'] = df['exang'].map({'True': 1, 'False': 0})
df['sex'] = df['sex'].str.capitalize()
```

```
In [11]: # 1. Histogram of patient counts by age group
bins = list(range(df['age'].min()//10*10, (df['age'].max()//10+1)*10+1, 10))
labels = [f"{bins[i]}-{bins[i+1]-1}" for i in range(len(bins)-1)]
df['age_group'] = pd.cut(df['age'], bins=bins, labels=labels, right=False)
plt.figure(figsize=(9,5))
sns.countplot(x='age_group', data=df, order=labels)
plt.title('Patient Count by Age Group')
```

```
plt.xlabel('Age Group')  
plt.ylabel('Count')  
plt.xticks(rotation=30)  
plt.tight_layout()  
plt.show()
```



```
In [12]: # 2. Filter: males aged 40-50  
filtered = df[(df['sex'] == 'Male') & (df['age'] >= 40) & (df['age'] <= 50)]  
print("Males aged 40-50:")  
print(filtered.head())
```

Males aged 40-50:

	id	age	sex	dataset	cp	trestbps	chol	fb	\
13	14	44	Male	Cleveland	atypical angina	120.0	263.0	NaN	
16	17	48	Male	Cleveland	atypical angina	110.0	229.0	NaN	
19	20	49	Male	Cleveland	atypical angina	130.0	266.0	NaN	
28	29	43	Male	Cleveland	asymptomatic	150.0	247.0	NaN	
29	30	40	Male	Cleveland	asymptomatic	110.0	167.0	NaN	

	restecg	thalch	exang	oldpeak	slope	ca	\
13	normal	173.0	NaN	0.0	upsloping	0.0	
16	normal	168.0	NaN	1.0	downsloping	0.0	
19	normal	171.0	NaN	0.6	upsloping	0.0	
28	normal	171.0	NaN	1.5	upsloping	0.0	
29	lv hypertrophy	114.0	NaN	2.0	flat	0.0	

	thal	num	age_group
13	reversible defect	0	40-49
16	reversible defect	1	40-49
19	normal	0	40-49
28	normal	0	40-49
29	reversible defect	3	40-49

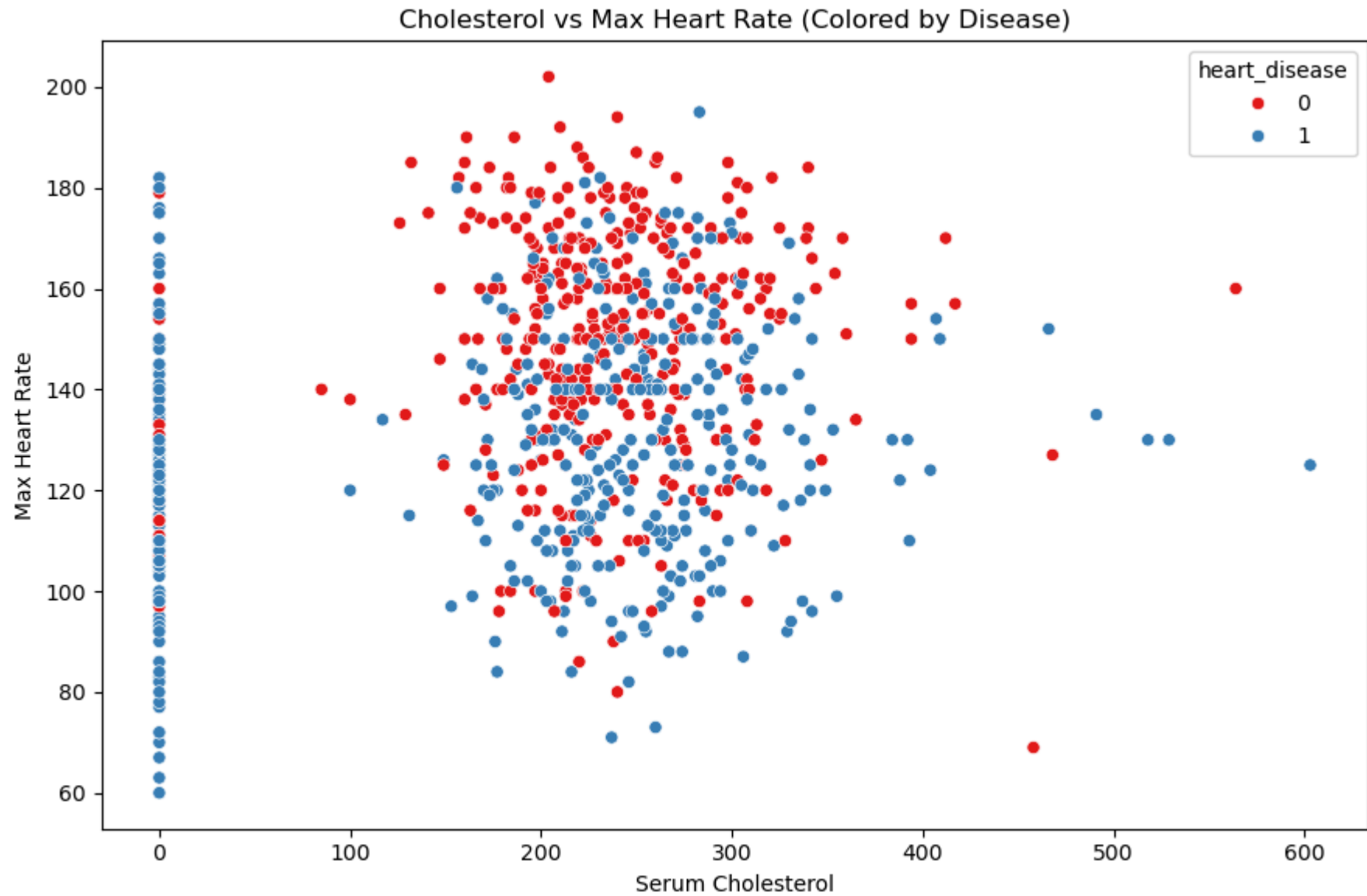
```
In [15]: # 3. Age group with highest heart disease percentage
# 'num' column: >0 means disease, 0 means no disease
df['heart_disease'] = (df['num'] > 0).astype(int)
age_group_hd_pct = df.groupby('age_group', observed=True)['heart_disease'].mean() * 100
print("\nHeart Disease % by Age Group:")
print(age_group_hd_pct)
print("Highest risk group:", age_group_hd_pct.idxmax(), f"({age_group_hd_pct.max():.2f}%)")
```

```
Heart Disease % by Age Group:
age_group
20-29      0.000000
30-39     34.210526
40-49     40.094340
50-59     56.800000
60-69     73.423423
70-79     70.967742
Name: heart_disease, dtype: float64
Highest risk group: 60-69 (73.42%)
```

```
In [16]: # 4. Top 3 major risk factors (by absolute correlation with heart_disease)
num_cols = ['age', 'trestbps', 'chol', 'thalch', 'oldpeak', 'ca']
corr_matrix = df[num_cols + ['heart_disease']].corr()['heart_disease'].abs().sort_values(ascending=False)
print("\nTop 3 numeric features correlated with heart disease:")
print(corr_matrix[1:4].index.tolist())
```

Top 3 numeric features correlated with heart disease:  
['ca', 'thalch', 'oldpeak']

```
In [17]: # 5. Scatter plot: cholesterol vs max heart rate, colored by heart disease status
plt.figure(figsize=(9,6))
sns.scatterplot(x='chol', y='thalch', hue='heart_disease', data=df, palette='Set1')
plt.title('Cholesterol vs Max Heart Rate (Colored by Disease)')
plt.xlabel('Serum Cholesterol')
plt.ylabel('Max Heart Rate')
plt.tight_layout()
plt.show()
```



```
In [18]: # 6. Compare average cholesterol: with and without heart disease
avg_chol = df.groupby('heart_disease')['chol'].mean()
print(f"\nAvg cholesterol (No Disease): {avg_chol.get(0, np.nan):.2f}")
print(f"Avg cholesterol (Disease): {avg_chol.get(1, np.nan):.2f}")
```

Avg cholesterol (No Disease): 227.91

Avg cholesterol (Disease): 176.48

```
In [19]: # 7. Diagnoses by month or time period
if 'month' in df.columns:
    print("Most diagnoses in month:", df['month'].value_counts().idxmax())
else:
    print("Month/time period data not available in this dataset.")
```

Month/time period data not available in this dataset.