Load dataset

```
import pandas as pd

# Working URL (from a stable repo - identical to Kaggle heart disease UCI)
url =
"https://raw.githubusercontent.com/sharmaroshan/Heart-UCI-Dataset/master/heart.csv"

# Load the data
df = pd.read_csv(url)

# Check if it loaded (should print shape (303, 14) and first rows)
print("Data loaded successfully!")
print("Shape:", df.shape)
print("\nColumns:", df.columns.tolist())
print("\nFirst 5 rows:")
print(df.head())
```

Main code:

'target' – heart disease yes/no)

```
# Add these imports (if not already there)
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error,
r2_score
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
# ——— LINEAR REGRESSION: Predict 'oldpeak' (continuous value) ———
print("\n" + "="*50)
print("LINEAR REGRESSION - Predicting oldpeak (ST depression)")
print("="*50)
```

```
X_lin = df.drop(['oldpeak', 'target'], axis=1)  # Features (drop what
we're predicting + binary target)
y_lin = df['oldpeak']  # Target for linear

X_train_lin, X_test_lin, y_train_lin, y_test_lin = train_test_split(
    X_lin, y_lin, test_size=0.2, random_state=42
)

lr = LinearRegression()
lr.fit(X_train_lin, y_train_lin)
y_pred_lin = lr.predict(X_test_lin)

print(f"MAE:  {mean_absolute_error(y_test_lin, y_pred_lin):.3f}")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test_lin, y_pred_lin)):.3f}")
print(f"R²:   {r2_score(y_test_lin, y_pred_lin):.3f}")

# Plot
plt.figure(figsize=(7,5))
plt.scatter(y_test_lin, y_pred_lin, alpha=0.6, color='blue')
plt.plot([y_test_lin.min(), y_test_lin.max()], [y_test_lin.min(),
y_test_lin.max()], 'r--')
plt.xlabel("Actual oldpeak")
plt.ylabel("Predicted oldpeak")
plt.title("Linear Regression: Actual vs Predicted")
plt.grid(True)
plt.show()

# ——— LOGISTIC REGRESSION: Predict 'target' (0=no disease, 1=disease) ———
print("\n" + "="*50)
print("LOGISTIC REGRESSION - Predicting Heart Disease")
print("="*50)

X_log = df.drop('target', axis=1)  # Features
y_log = df['target']  # Binary target

X_train_log, X_test_log, y_train_log, y_test_log = train_test_split(
    X_log, y_log, test_size=0.2, random_state=42, stratify=y_log
)

log_model = LogisticRegression(max_iter=2000, random_state=42)
```

```
log_model.fit(X_train_log, y_train_log)
y_pred_log = log_model.predict(X_test_log)

acc = accuracy_score(y_test_log, y_pred_log)
print(f"Accuracy: {acc:.3f} ({acc*100:.1f}%)")

print("\nConfusion Matrix:")
print(confusion_matrix(y_test_log, y_pred_log))

print("\nClassification Report:")
print(classification_report(y_test_log, y_pred_log))

# Heatmap plot
plt.figure(figsize=(6,5))
sns.heatmap(confusion_matrix(y_test_log, y_pred_log), annot=True, fmt='d',
cmap='Greens', cbar=False)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - Heart Disease')
plt.show()
```