import matplotlib.pyplot as plt

import numpy as np

# Generate x values from 0 to 2π with 100 points

x = np.linspace(0, 2\*np.pi, 100)

# Calculate sine values for each x

y = np.sin(x)

# Plot the sine wave

plt.plot(x, y)

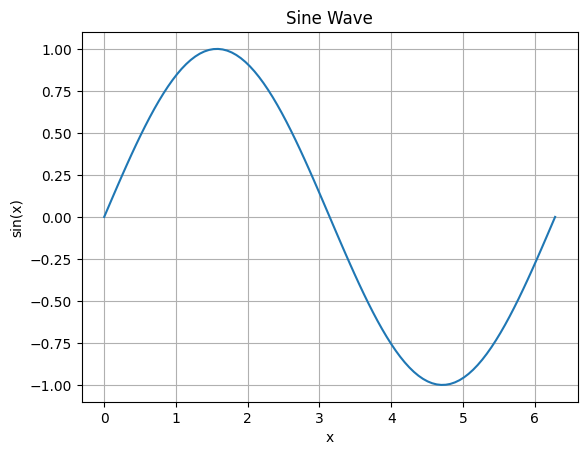
plt.xlabel('x')

plt.ylabel('sin(x)')

plt.title('Sine Wave')

plt.grid(True)

plt.show()



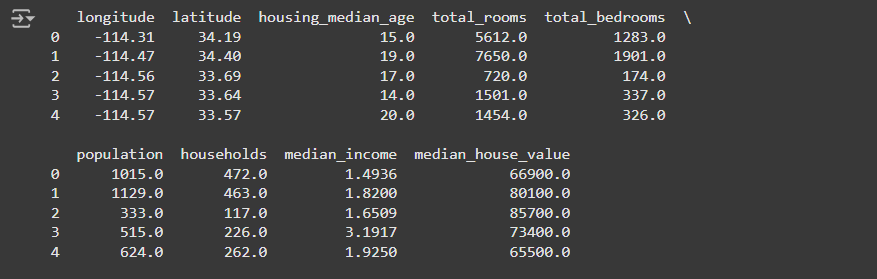
import pandas as pd

# Load the dataset from a CSV file

df = pd.read\_csv('/content/sample\_data/california\_housing\_train.csv')

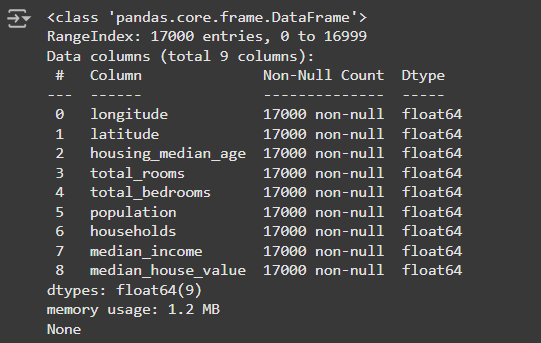
# Display the first 5 rows

print(df.head())



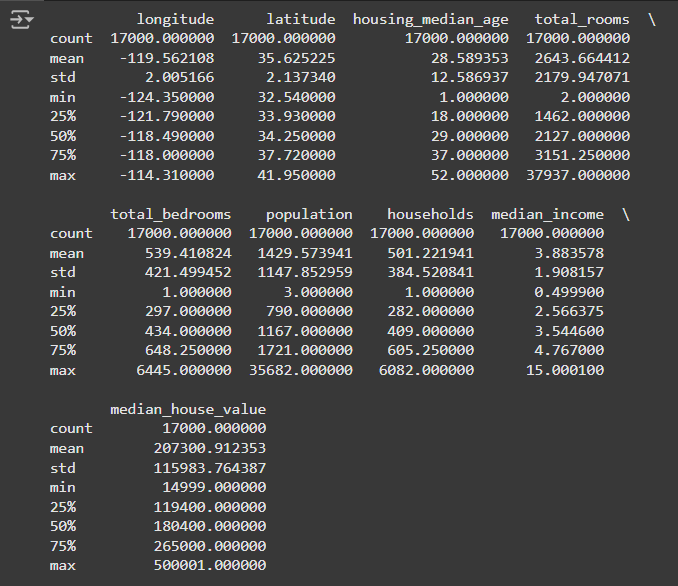
# Get information about the dataset

print(df.info())



# Calculate basic statistics

print(df.describe())



**A Simple Machine Learning Pipeline in Google Colab**

A basic machine learning pipeline demonstrating how to train a simple linear regression model on a dataset

import pandas as pd

import numpy as np

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

# Load the dataset

data = {'X': [1, 2, 3, 4, 5], 'y': [2, 4, 5, 4, 5]}

df = pd.DataFrame(data)

# Split data into features (X) and target variable (y)

X = df[['X']]

y = df['y']

# Split data into training and testing sets

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

# Create a linear regression model

model = LinearRegression()

# Train the model on the training data

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

# Make predictions on the testing data

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

# Evaluate the model's performance

mse = mean\_squared\_error(y\_test, y\_pred)

print("Mean Squared Error:", mse)

