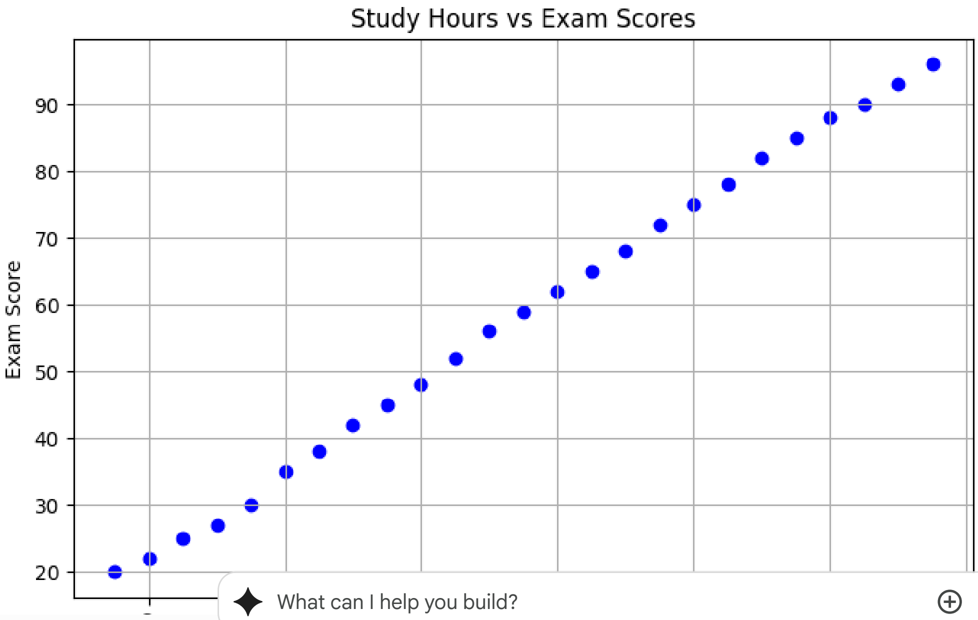
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error


# large data set
data = {
    'Hours': [1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0,
              6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5, 10.0, 10.5, 11.0,
              11.5, 12.0, 12.5, 13.0, 13.5],
    'Scores': [20, 22, 25, 27, 30, 35, 38, 42, 45, 48,
               52, 56, 59, 62, 65, 68, 72, 75, 78, 82,
               85, 88, 90, 93, 96]
}

# Creating a  DataFrame
df = pd.DataFrame(data)

# Show the first few rows
df.head()
```

	Hours	Scores
0	1.5	20
1	2.0	22
2	2.5	25
3	3.0	27
4	3.5	30

Next steps: Generate code with df View recommended plots New interactive sheet

```
# Scatter plot
plt.figure(figsize=(8,5))
plt.scatter(df['Hours'], df['Scores'], color='blue', marker='o')
plt.title('Study Hours vs Exam Scores')
plt.xlabel('Hours Studied')
plt.ylabel('Exam Score')
plt.grid(True)
plt.show()
```



What can I help you build?

```
# Make predictions on test data
```

```
y_pred = model.predict(X_test)

# Compare predicted and actual values
results = pd.DataFrame({'Hours Studied': X_test['Hours'], 'Actual Score': y_test, '
print(results)

# Calculate Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
print(f"\nMean Squared Error: {mse:.2f}")
```

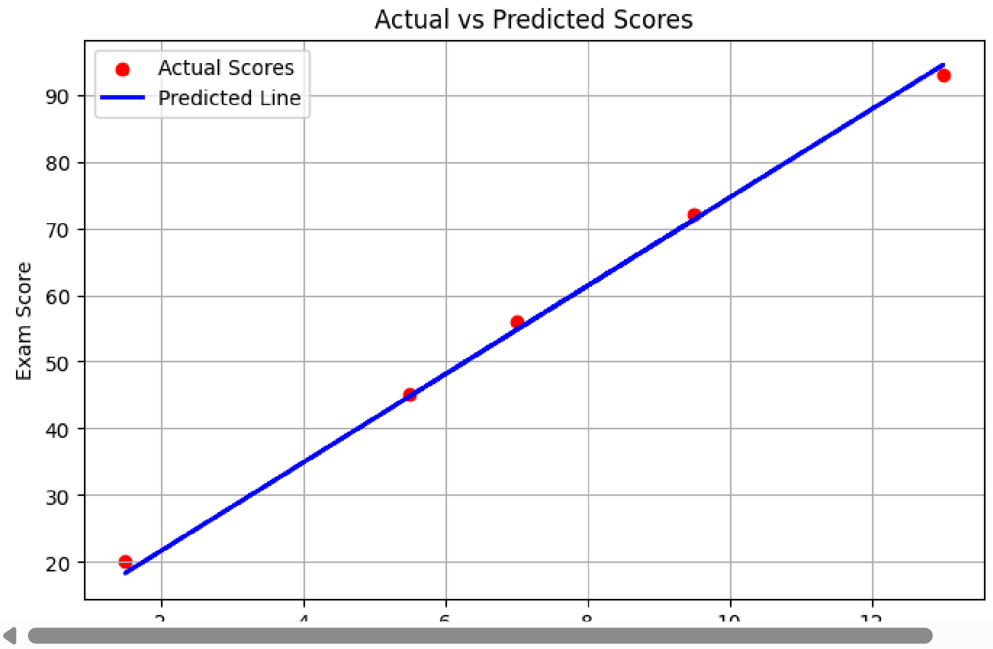
```
    Hours Studied  Actual Score  Predicted Score
8             5.5            45        44.785786
16            9.5            72        71.252545
0             1.5            20        18.319026
23           13.0            93        94.410960
11            7.0            56        54.710821

Mean Squared Error: 1.42
```

Start coding or generate with AI.

```
# Split the data into inputs and outputs
X = df[['Hours']]  # Features (input)
y = df['Scores']   # Target (output)

# Split into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_s

# Create and train the Linear Regression model
model = LinearRegression()
model.fit(X_train, y_train)
```

```
▾ LinearRegression  ⓘ ?
LinearRegression()
```

```
# Plotting Actual vs Predicted
plt.figure(figsize=(8,5))
plt.scatter(X_test, y_test, color='red', label='Actual Scores')
plt.plot(X_test, y_pred, color='blue', linewidth=2, label='Predicted Line')
plt.title('Actual vs Predicted Scores')
plt.xlabel('Hours Studied')
plt.ylabel('Exam Score')
plt.legend()
plt.grid(True)
plt.show()
```



```
## Student Performance Prediction Using
Linear Regression

**Tools Used:** Python, Pandas,
Matplotlib, Scikit-learn, Google Colab

**Description:**
Built a machine learning model to
predict student scores based on their
study hours using linear regression. The
project involved data preprocessing,
visualization, model training,
evaluation (MSE), and visualization of
predictions. Achieved good accuracy with
a clearly interpretable model.
```

Student Performance Prediction Using Linear Regression

Tools Used: Python, Pandas, Matplotlib, Scikit-learn, Google Colab

Description:
Built a machine learning model to predict student scores based on their study hours using linear regression. The project involved data preprocessing, visualization, model training, evaluation (MSE), and visualization of predictions. Achieved good accuracy with a clearly interpretable model.