**Scenario:** Suppose we have collected data on the number of hours students studied for an exam (x) and their corresponding scores (y). We fit a linear regression model to predict the exam score based on study hours. Our model is:

y^​=50+7x

We have the following actual scores and the scores predicted by our model for 5 students:

|  |  |  |  |
| --- | --- | --- | --- |
| **Student** | **Study Hours (x)** | **Actual Score (y)** | **Predicted Score (y^​=50+7x)** |
| 1 | 2 | 62 | 50+7(2)=64 |
| 2 | 3 | 75 | 50+7(3)=71 |
| 3 | 4 | 80 | 50+7(4)=78 |
| 4 | 5 | 88 | 50+7(5)=85 |
| 5 | 6 | 91 | 50+7(6)=92 |

Export to Sheets

Now, let's calculate the different error and accuracy metrics:

**1. Mean Absolute Error (MAE):**

First, we calculate the absolute errors for each student:

* ∣62−64∣=2
* ∣75−71∣=4
* ∣80−78∣=2
* ∣88−85∣=3
* ∣91−92∣=1

Now, we take the average of these absolute errors: MAE=52+4+2+3+1​=512​=2.4

**Interpretation:** On average, our model's predictions are off by 2.4 points from the actual exam scores.

**2. Mean Squared Error (MSE):**

Next, we calculate the squared errors for each student:

* (62−64)2=(−2)2=4
* (75−71)2=(4)2=16
* (80−78)2=(2)2=4
* (88−85)2=(3)2=9
* (91−92)2=(−1)2=1

Now, we take the average of these squared errors: MSE=54+16+4+9+1​=534​=6.8

**Interpretation:** The average of the squared errors is 6.8. Notice this value is larger than the MAE because the squaring process amplifies the larger errors.

**3. Root Mean Squared Error (RMSE):**

RMSE is the square root of the MSE: RMSE=6.8​≈2.61

**Interpretation:** The RMSE is approximately 2.61 points. This is in the same unit as the exam scores, making it a bit more interpretable than MSE. It tells us the standard deviation of the prediction errors.

**4. R-squared (R²):**

To calculate R-squared, we need the Total Sum of Squares (SST) and the Sum of Squared Errors (SSE). We already calculated SSE (which is the numerator in the MSE calculation, multiplied by n): SSE=i=1∑5​(yi​−y^​i​)2=34

Now, let's calculate the mean of the actual scores (yˉ​): yˉ​=562+75+80+88+91​=5396​=79.2

Now, we calculate the SST:

* (62−79.2)2=(−17.2)2=295.84
* (75−79.2)2=(−4.2)2=17.64
* (80−79.2)2=(0.8)2=0.64
* (88−79.2)2=(8.8)2=77.44
* (91−79.2)2=(11.8)2=139.24

SST=295.84+17.64+0.64+77.44+139.24=530.8

Finally, we can calculate R-squared: R2=1−SSTSSE​=1−530.834​=1−0.064=0.936

**Interpretation:** An R-squared of 0.936 (or 93.6%) indicates that our linear regression model explains approximately 93.6% of the variance in the exam scores based on the number of study hours. This suggests a strong fit of the model to the data.

**5. Adjusted R-squared:**

For adjusted R-squared, we have n=5 data points and p=1 predictor (study hours). Radj2​=1−n−p−1(1−R2)(n−1)​=1−5−1−1(1−0.936)(5−1)​=1−3(0.064)(4)​=1−30.256​=1−0.0853≈0.915

**Interpretation:** The adjusted R-squared is approximately 0.915. Since we only have one predictor, the adjusted R-squared is quite close to the regular R-squared. If we had added more potentially irrelevant predictors, the adjusted R-squared would likely be lower than the R-squared, reflecting the penalty for those additional variables.

This example illustrates how to calculate and interpret these common metrics for evaluating the

**Which method is good?**

The "best" method depends on the specific context and goals:

* **MAE:** Useful when you want a simple, easily interpretable metric in the original units and are less concerned about the impact of outliers.
* **MSE:** Important when large errors should be heavily penalized, often used as the loss function in optimization algorithms.
* **RMSE:** Provides a good balance between the interpretability of MAE and the error penalization of MSE. It's a widely used metric.
* **R-squared:** Offers a general sense of how well the model fits the data by explaining the variance. However, it doesn't tell you about the magnitude of errors and can be misleading if the model overfits or if irrelevant variables are added.
* **Adjusted R-squared:** More suitable for multiple linear regression as it helps in selecting the most relevant predictors and avoids overfitting by penalizing the inclusion of unnecessary variables.

In practice, it's often beneficial to look at multiple metrics to get a comprehensive understanding of your linear regression model's performance. For example, a model might have a high R-squared but also a high RMSE, indicating a good overall fit but with some large errors.

import numpy as np

from sklearn.linear\_model import LinearRegression

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

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

import math

# Sample data (similar to our previous example, but let's make it a bit more realistic)

X = np.array([[2], [3], [4], [5], [6], [7], [8]]) # Study hours

y = np.array([60, 72, 78, 86, 90, 95, 98]) # Exam scores

# Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# Train a linear regression model

model = LinearRegression()

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

# Make predictions on the test set

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

# Calculate MAE

mae = mean\_absolute\_error(y\_test, y\_pred)

print(f"Mean Absolute Error (MAE): {mae:.2f}")

# Calculate MSE

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

print(f"Mean Squared Error (MSE): {mse:.2f}")

# Calculate RMSE

rmse = math.sqrt(mse)

print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")

# Calculate R-squared

r\_squared = r2\_score(y\_test, y\_pred)

print(f"R-squared (R²): {r\_squared:.2f}")

# Calculate Adjusted R-squared

n = len(y\_test) # Number of data points in the test set

p = X\_test.shape[1] # Number of features (which is 1 in this simple linear regression)

adjusted\_r\_squared = 1 - (1 - r\_squared) \* (n - 1) / (n - p - 1)

print(f"Adjusted R-squared: {adjusted\_r\_squared:.2f}")