What is Mean Squared Error (MSE)?

Mean Squared Error (MSE) is a **regression metric** that measures the **average squared difference** between predicted and actual values. It quantifies how far predictions deviate from the true values, with higher errors penalized more severely due to squaring.

Mathematical Formula

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- y_i: True value
- $\hat{y}_i$: Predicted value
- n: Number of samples

Key Properties

1. **Always Non-Negative**:
   Since errors are squared, MSE ≥ 0.
2. **Units**:
   MSE is in **squared units** of the target variable (e.g., if y is in meters, MSE is in m^2).
3. **Sensitive to Outliers**:
   Large errors are heavily penalized due to squaring.

Example Calculation

y (Actual)	$\hat{y}$ (Predicted)	Error $(y - \hat{y})$	Squared Error $(y - \hat{y})^2$
3	2.5	0.5	0.25
5	5	0	0
2.5	4	-1.5	2.25
7	8	-1	1

$$\text{MSE} = \frac{0.25 + 0 + 2.25 + 1}{4} = \frac{3.5}{4} = 0.875$$

Python Implementation

Using `sklearn.metrics`:

```python
from sklearn.metrics import mean_squared_error

y_true = [3, 5, 2.5, 7]
y_pred = [2.5, 5, 4, 8]

mse = mean_squared_error(y_true, y_pred)
print(f"MSE: {mse}")  # Output: MSE: 0.875
```

Manual Calculation (NumPy):

```python
import numpy as np

y_true = np.array([3, 5, 2.5, 7])
y_pred = np.array([2.5, 5, 4, 8])

squared_errors = (y_true - y_pred) ** 2
mse = np.mean(squared_errors)
print(f"MSE: {mse}")  # Output: MSE: 0.875
```

When to Use MSE

- **Regression Problems**: Predicting continuous values (e.g., house prices, temperature).
- **Model Evaluation**: Compare performance of different regression models.
- **Loss Function**: Often used as the optimization objective in algorithms like linear regression.

Pros and Cons

Pros	Cons
Easy to interpret and compute.	Sensitive to outliers.
Differentiable (useful for training).	Units are not intuitive (squared).
Emphasizes large errors.	Hard to compare across datasets with different scales.

Related Metrics

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

$$\text{RMSE} = \sqrt{\text{MSE}}$$

   - Converts MSE back to original units.
   - More interpretable but still sensitive to outliers.

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

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

   - Robust to outliers but not differentiable.

Example with MLflow Logging

python Copy

```python
import mlflow
from sklearn.linear_model import LinearRegression
from sklearn.datasets import fetch_california_housing
from sklearn.model_selection import train_test_split

# Load data
data = fetch_california_housing()
X_train, X_test, y_train, y_test = train_test_split(data.data, data.target)

# Train model
model = LinearRegression().fit(X_train, y_train)
y_pred = model.predict(X_test)

# Log MSE with MLflow
with mlflow.start_run():
    mse = mean_squared_error(y_test, y_pred)
    mlflow.log_metric("mse", mse)
    mlflow.sklearn.log_model(model, "model", input_example=X_test[:1])
```

Key Takeaways

- MSE measures **average squared prediction error**.
- Use `sklearn.metrics.mean_squared_error` for quick calculation.

- Prefer RMSE or MAE if outliers or interpretability are concerns.
- Always log MSE (or other metrics) in MLflow for tracking model performance.