Lesson 23: Regression Metrics

Data Science with Python – BSc Course

Data Science Program

BSc Course

45 Minutes

Previously on L21–L22...

L21 – Linear Regression:

- OLS finds the best-fit line by minimizing squared errors
- β = slope, α = intercept

L22 – Regularization:

- Ridge shrinks coefficients to prevent overfitting
- Lasso selects features by driving some coefficients to zero

But how do we KNOW our model is any good?

We can fit a line. We can regularize it. But what evidence do we have that it actually *works*?

The missing piece: objective tools to measure and compare model quality

Learning Objectives

The Problem: We've built a regression model, but how good is it? How do we compare different models objectively?

After this lesson, you will be able to:

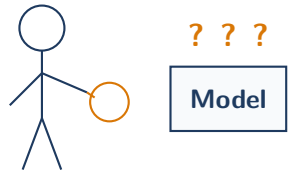
- Calculate MSE, RMSE, and MAE for prediction quality
- Interpret R^2 as variance explained
- Use adjusted R^2 to penalize model complexity
- Apply time series cross-validation for financial data

Finance Application: Evaluating return prediction models before deployment

The Trust Problem

Your model says it's 95% accurate. On what data?

The data it already saw. You wouldn't trust a student who graded their own exam.



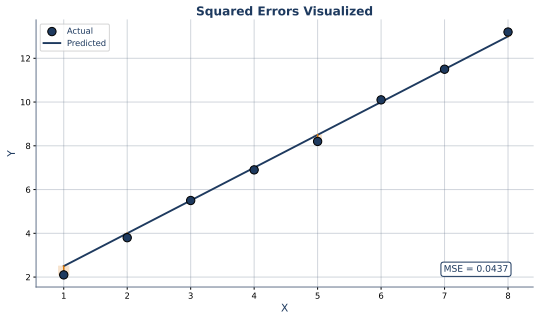
We need **independent evidence**. Today: the detective toolkit for interrogating models.

Like forensic accounting – we need tools to verify claims, not just trust them

Measuring Wrongness: MSE

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

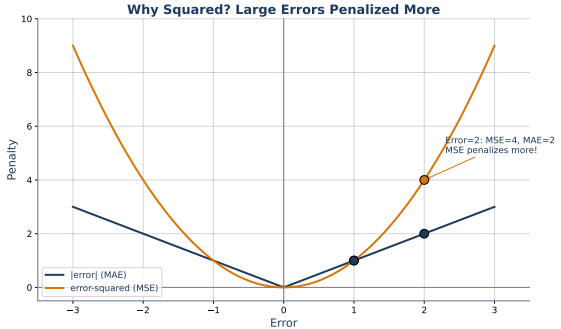
Why squares? Always positive, penalizes outliers, has closed-form solution.



MSE = average of squared residuals – penalizes large errors heavily

MSE vs MAE: Two Philosophies

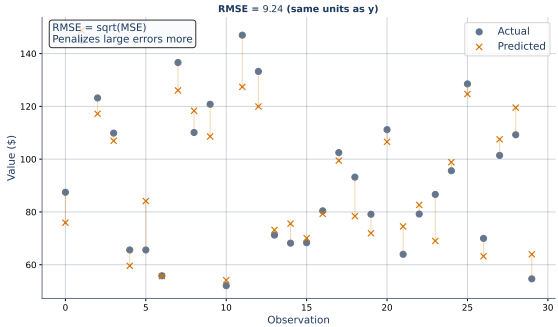
- MSE: squaring makes a 10% error $\rightarrow$ 100 penalty units
- MAE: absolute value makes a 10% error $\rightarrow$ 10 penalty units



Squaring makes MSE sensitive to outliers – one large error dominates

RMSE: Error in Real Units

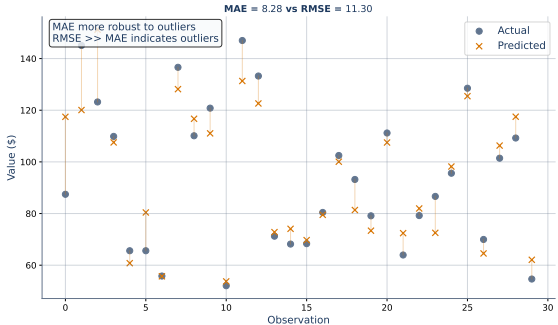
- RMSE $= \sqrt{\text{MSE}}$ – same units as y
- In finance: a 10% prediction error could cost millions



Use RMSE when large errors are particularly costly

MAE: The Robust Alternative

- MAE $= \frac{1}{n}\sum |y_i - \hat{y}_i|$ – treats all errors equally
- Not affected by outliers; use when outliers may be noise



Use MAE when outliers may be noise or data quality issues

Worked Example: Calculating Metrics

Data:

- Predictions: [10, 12, 15] Actuals: [11, 11, 16] Errors: [-1, +1, -1]

Calculations:

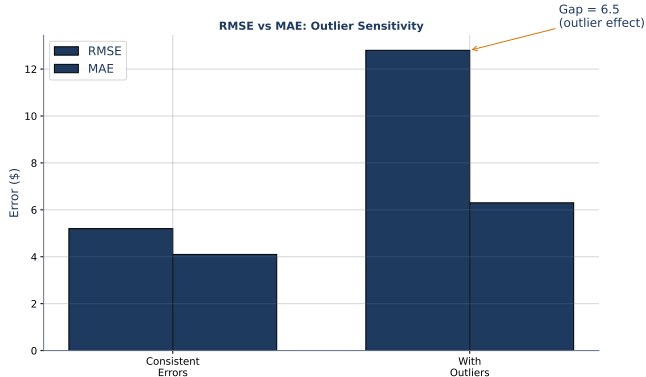
- **MAE** $= \frac{|-1|+|1|+|-1|}{3} = \frac{3}{3} = 1.0$ dollars
- **MSE** $= \frac{(-1)^2+(1)^2+(-1)^2}{3} = \frac{3}{3} = 1.0$ dollars^2
- **RMSE** $= \sqrt{1.0} = 1.0$ dollars

Interpretation:

- "On average, predictions off by about $1"
- MAE = RMSE here because all errors have same magnitude
- If one error were larger, RMSE $>$ MAE

Rule of thumb: If RMSE ¿¿ MAE, you have outlier predictions to investigate

RMSE vs MAE Comparison

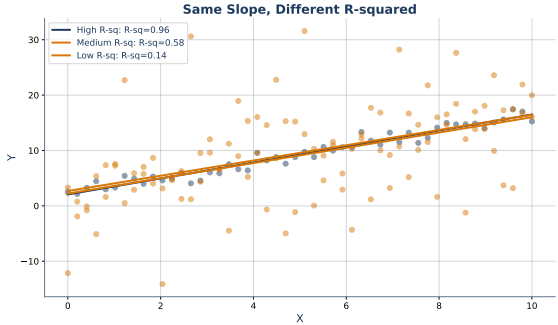


Rule: If RMSE ¿¿ MAE, you have outliers. Investigate before choosing metric.

R-Squared: The Headline Number

What fraction of variance does our model explain?

$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$ ($R^2 = 0$: nothing explained; $R^2 = 1$: perfect)

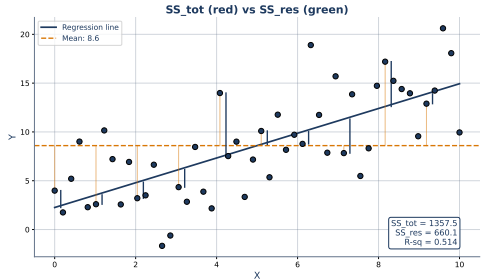


Higher R-squared = tighter fit around regression line

Building R-Squared from SS Components

- **SS_tot**: $\sum(y_i - \bar{y})^2$ – total variation
- **SS_res**: $\sum(y_i - \hat{y}_i)^2$ – leftover after predictions
- **SS_reg**: $\sum(\hat{y}_i - \bar{y})^2$ – explained by model

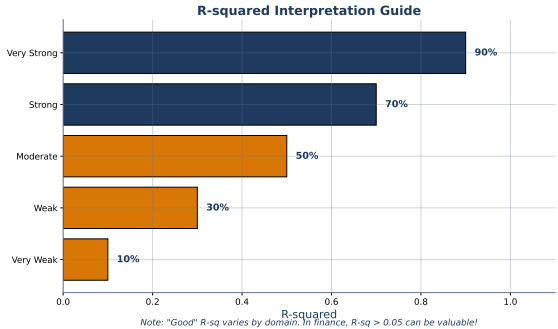
$$\text{SS\_tot} = \text{SS\_reg} + \text{SS\_res}$$



R^2 = **SS_reg / SS_tot = fraction of variation explained by model**

R-Squared: Context Matters

- $R^2 = 0.3$ is **excellent** for finance (returns are noisy)
- $R^2 = 0.3$ is **poor** for physics (precise laws)

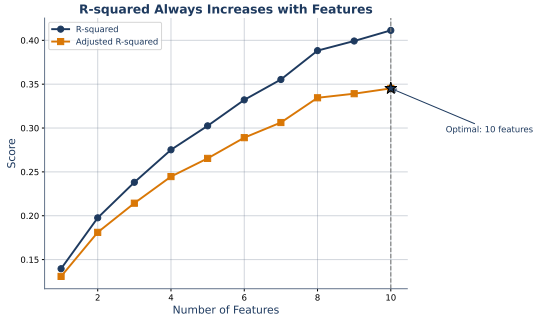


Context matters: R^2=0.3 may be excellent for returns, poor for physics

The Adjusted R-Squared Trap

Adding ANY feature raises R^2 – even noise! Adjusted R^2 penalizes this.

Adj $R^2 = 1 - \frac{(1-R^2)(n-1)}{n-p-1}$ Rule: keep feature only if adj R^2 increases.



Regular R^2 always increases with features; adjusted penalizes complexity

Residual Forensics: The Four Plots

Run all four before trusting regression results:

Residuals vs Fitted
Check for patterns (nonlinearity)

QQ Plot
Check normality (heavy tails?)

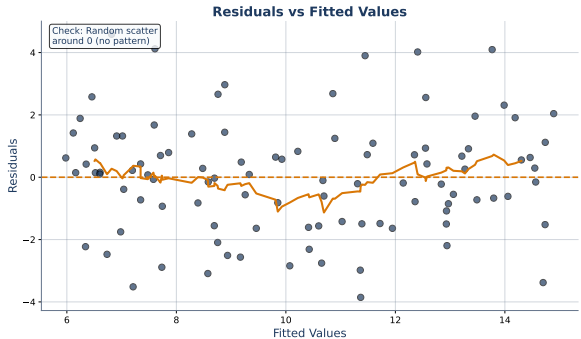
Scale-Location
Check equal variance

Residuals vs Order
Check time independence

Like a medical checkup – four tests that together reveal hidden model problems

Residual Plot 1: Patterns = Problems

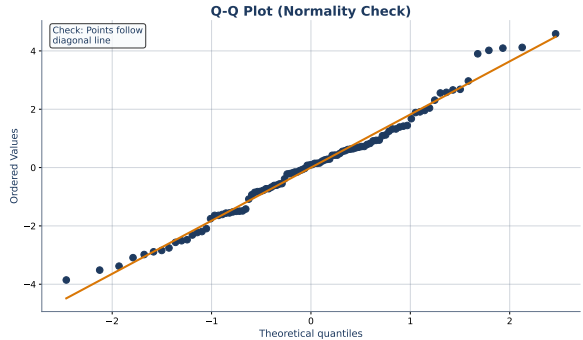
- Random scatter = good; funnel = heteroscedasticity; curve = nonlinearity



Random scatter around zero = your model captures the signal correctly

Residual Plot 2: QQ for Normality

- Diagonal = normal; S-curve = heavy tails; banana = skewed



Points on diagonal = normal residuals; deviations show skew or heavy tails

Checkpoint: Reading Residual Plots

Quick Quiz:

Q1: "You see a funnel-shaped residual plot. What does this tell you?"

A: Heteroscedasticity – variance increases with fitted values.

Fixes: log-transform Y, weighted least squares, robust standard errors.

Q2: "You see an S-curve in the QQ plot. What does it mean?"

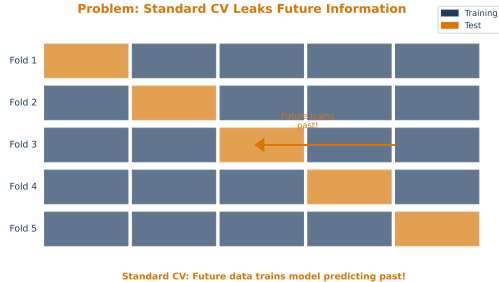
A: Heavy tails – more extreme values than a normal distribution predicts.

Action: check for outliers, consider robust regression.

If you can read residual plots, you can diagnose most regression problems

Time Series CV: The Data Leakage Trap

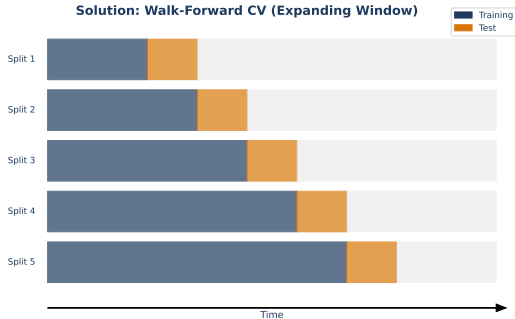
- Standard CV shuffles data → future leaks into training set
- Training on 2024 data to predict 2023 = "cheating"



Standard CV shuffles time order → future leaks into past → overly optimistic

Walk-Forward Validation

- Train on past, test on future; window expands forward
- sklearn: `TimeSeriesSplit(n_splits=5)`



Walk-forward: always train on past, test on future – no leakage

Common Mistakes in Model Evaluation

Four errors that mislead you:

1. **Evaluating on training data only**
   Training R^2 is always optimistic – test on held-out data
2. **Using standard K-fold CV for time series**
   Future leaks into past $\rightarrow$ use walk-forward validation
3. **Ignoring residual patterns**
   "It fits, ship it!" $\rightarrow$ always check diagnostic plots
4. **Reporting R^2 without domain context**
   $R^2 = 0.3$ is NOT bad in finance – returns are inherently noisy!

Most model evaluation failures come from these four mistakes

Hands-On Exercise (25 min)

Task: Evaluate a Stock Return Prediction Model

1. Fit a linear regression predicting next-day returns from lagged features
2. Calculate MSE, RMSE, MAE, and R^2 on a test set
3. Plot residuals vs predicted values – any patterns?
4. Use `TimeSeriesSplit` for proper cross-validation

Deliverable: Summary table of metrics + residual plot.

Reference: `inclass/L23_inclass_metric_pitfalls.ipynb`

Extension: Calculate the RMSE/MAE ratio – does it suggest outlier issues?

Lesson Summary + Preview

Problem Solved: Objectively measure and compare regression model quality.

Key Takeaways:

- RMSE/MAE: interpretable error in original units
- R^2: proportion of variance explained (0 to 1)
- Adjusted R^2: penalizes unnecessary complexity
- Walk-forward validation: prevents data leakage in time series

Next Lesson: *"One factor explains 60% of stock returns. What about the other 40%?"*

Next: **L24 – Factor Models**: One Factor Is Never Enough.

Memory: RMSE punishes outliers, MAE treats all equally, R^2 = % variance explained