Introduction & Linear Regression

Deep Dive: Mathematics and Implementation

Methods and Algorithms

MSc Data Science

Spring 2026

Outline

Learning Objectives

By the end of this session, you will be able to:

1. **Derive** the OLS estimator and prove its optimality under Gauss-Markov assumptions
2. **Analyze** gradient descent convergence and evaluate learning rate selection
3. **Evaluate** regression diagnostics to identify assumption violations
4. **Compare** regularization strategies (Ridge, Lasso, Elastic Net) for different problem structures

Finance Applications: Property valuation, asset pricing (CAPM)

Foundation for all supervised learning methods

The Business Challenge

Finance Use Case: House Price Prediction

- Banks need accurate property valuations for mortgage decisions
- Insurers must estimate replacement costs
- Investors evaluate real estate portfolios

Why Linear Regression?

- Interpretable coefficients (how much does each feature matter?)
- Regulatory requirement for explainable models
- Fast, well-understood baseline

Linear regression: the workhorse of quantitative finance since the 1800s

Finance Application: The CAPM

Capital Asset Pricing Model – Linear Regression in Finance

$$R_i - R_f = \alpha_i + \beta_i(R_m - R_f) + \varepsilon_i \tag{1}$$

- R_i: Return of asset i
- R_f: Risk-free rate (e.g., T-bill)
- R_m: Market return (e.g., S&P 500)
- β_i: Systematic risk (market sensitivity)
- α_i: Abnormal return (should be zero if CAPM holds)

Interpretation: $\beta = 1.2$ means 10% market rise $\Rightarrow$ 12% expected asset rise

CAPM: The original factor model – basis for portfolio management

Matrix Notation

The Model in Matrix Form

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{2}$$

- $\mathbf{y} \in \mathbb{R}^n$: Response vector
- $\mathbf{X} \in \mathbb{R}^{n\times(p+1)}$: Design matrix (with intercept column)
- $\boldsymbol{\beta} \in \mathbb{R}^{p+1}$: Coefficient vector
- $\boldsymbol{\varepsilon} \in \mathbb{R}^n$: Error vector

Matrix notation enables elegant derivations and efficient computation

Design Matrix Structure

The Design Matrix $\mathbf{X}$

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1p} \\ 1 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix} \tag{3}$$

- First column of 1s for intercept β_0
- Each row is one observation
- Each column (after first) is one feature

n **observations,** p **features,** $p+1$ **parameters**

OLS Assumptions

Classical Assumptions for Valid Inference

1. **Linearity**: $E[y|X] = X\beta$ (correct functional form)
2. **Exogeneity**: $E[\varepsilon|X] = 0$ (no omitted variables)
3. **Homoscedasticity**: $\text{Var}(\varepsilon|X) = \sigma^2 I$ (constant variance)
4. **No multicollinearity**: $\text{rank}(X) = p + 1$ (full rank)
5. **Normality** (for inference): $\varepsilon \sim N(0, \sigma^2 I)$

Violations? Robust standard errors, transformations, regularization

---

Assumptions 1-4 needed for unbiased estimates; 5 for t-tests and CIs

The Gauss-Markov Theorem

Why OLS is Special

Under assumptions 1-4 (linearity, exogeneity, homoscedasticity, no perfect multicollinearity):

OLS is BLUE – Best Linear Unbiased Estimator

What BLUE Means:

- **Best**: Lowest variance among all linear unbiased estimators
- **Linear**: Estimator is linear function of y
- **Unbiased**: $E[\hat{\beta}] = \beta$

Implication: You cannot find a better linear unbiased estimator than OLS

Gauss-Markov justifies why OLS is the default choice for linear regression

Gauss-Markov: Proof Sketch

Goal: Show OLS $\hat{\boldsymbol{\beta}}$ has minimum variance among all linear unbiased estimators.
Proof: Let $\tilde{\boldsymbol{\beta}} = \mathbf{C}\mathbf{y}$ be any other linear unbiased estimator. Write $\mathbf{C} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' + \mathbf{D}$ for some matrix $\mathbf{D}$.

- Unbiasedness requires $E[\tilde{\boldsymbol{\beta}}] = \boldsymbol{\beta}$, which forces $\mathbf{D}\mathbf{X} = \mathbf{0}$
- Then: $\text{Var}(\tilde{\boldsymbol{\beta}}) = \sigma^2(\mathbf{X}'\mathbf{X})^{-1} + \sigma^2\mathbf{D}\mathbf{D}'$
- Since $\mathbf{D}\mathbf{D}' \succeq \mathbf{0}$ (positive semi-definite), we have:

$$\text{Var}(\tilde{\boldsymbol{\beta}}) - \text{Var}(\hat{\boldsymbol{\beta}}) = \sigma^2\mathbf{D}\mathbf{D}' \succeq \mathbf{0} \tag{4}$$

Therefore OLS has the smallest variance among all linear unbiased estimators. $\square$

The key insight: any deviation D from OLS adds noise ($\mathbf{DD}'$) without reducing bias.

The Loss Function

Sum of Squared Residuals (SSR)

$$L(\boldsymbol{\beta}) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \tag{5}$$

Expanding:

$$L(\boldsymbol{\beta}) = \mathbf{y}^\top\mathbf{y} - 2\boldsymbol{\beta}^\top\mathbf{X}^\top\mathbf{y} + \boldsymbol{\beta}^\top\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} \tag{6}$$

Quadratic function in β – has unique minimum (if X full rank)

Deriving the Normal Equation

Taking the Derivative

$$\frac{\partial L}{\partial \boldsymbol{\beta}} = -2\mathbf{X}^\top \mathbf{y} + 2\mathbf{X}^\top \mathbf{X}\beta \tag{7}$$

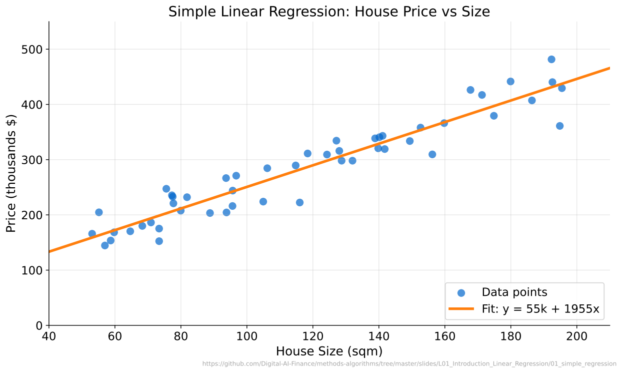
Setting to Zero:

$$\mathbf{X}^\top \mathbf{X}\hat{\beta} = \mathbf{X}^\top \mathbf{y} \tag{8}$$

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top \mathbf{y} \tag{9}$$

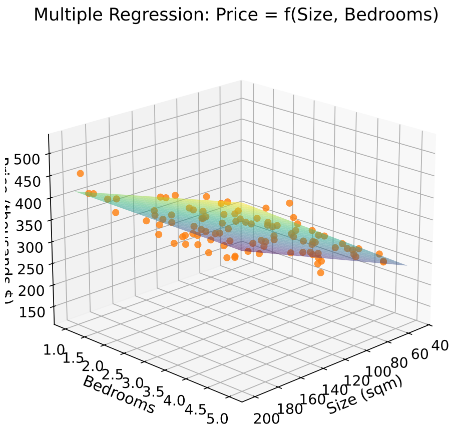
This is the closed-form OLS solution – the “normal equation”

Simple Regression Visualization



The fitted line minimizes vertical distances squared

Multiple Regression Surface



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L01_Introduction_Linear_Regression/02_multiple_regression_3d

With 2 features, we fit a plane; with *p* features, a hyperplane

Feature Scaling

Standardization: Zero Mean, Unit Variance

$$x_j^{\text{scaled}} = \frac{x_j - \bar{x}_j}{s_j} \tag{10}$$

Why Scale Features?

1. **Coefficient comparison:** After scaling, $|\beta_j|$ reflects relative importance
2. **Gradient descent:** Converges faster with similar feature scales
3. **Regularization:** Fair penalty across all features

Caution: Standardized coefficients lose original units (trade-off)

Always standardize for regularization; optional for OLS if only predicting

Why Gradient Descent?

Normal Equation Limitations

- Computing $(\mathbf{X}^\top \mathbf{X})^{-1}$ is $O(p^3)$
- Memory: Need to store $p \times p$ matrix
- For large p (millions of features): infeasible

Gradient Descent Advantages

- Memory efficient: process one sample at a time
- Scales to big data (SGD)
- Generalizes to non-linear models

For $p > 10{,}000$, gradient descent usually faster

The Gradient

Gradient of the Loss Function

$$\nabla L(\boldsymbol{\beta}) = -2\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = -2\mathbf{X}^\top\mathbf{r} \tag{11}$$

where $\mathbf{r} = \mathbf{y} - \mathbf{X}\boldsymbol{\beta}$ is the residual vector.

Intuition:

- Gradient points in direction of steepest ascent
- We move opposite to gradient (steepest descent)
- Scale by learning rate α

Gradient is a $p+1$ dimensional vector

Gradient Descent Algorithm

Update Rule

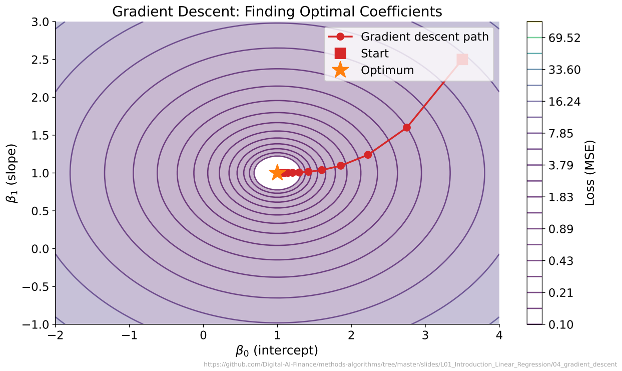
$$\beta^{(t+1)} = \beta^{(t)} - \alpha \nabla L(\beta^{(t)}) \tag{12}$$

Algorithm:

1. Initialize $\beta^{(0)}$ (often zeros or random)
2. Compute gradient $\nabla L(\beta^{(t)})$
3. Update: $\beta^{(t+1)} = \beta^{(t)} - \alpha \nabla L$
4. Repeat until convergence

Convergence: $\|\beta^{(t+1)} - \beta^{(t)}\| < \epsilon$ **or max iterations**

Gradient Descent Visualization



Contours show loss surface; path shows optimization trajectory

Learning Rate Selection

The Critical Hyperparameter

- **Too small**: Slow convergence, many iterations
- **Too large**: Divergence, oscillation
- **Just right**: Fast, stable convergence

Convergence Theory:

- Convex: $O(1/t)$ convergence rate
- Strongly convex: $O(\rho^t)$ where $\rho < 1$ (linear rate)
- Learning rate condition: $\eta < 2/L$ where L is Lipschitz constant of ∇L

Practical: Start with $\eta = 0.01$, use adaptive methods (Adam, AdaGrad)

For OLS, optimal $\eta = 1/\lambda_{\max}(X^\top X)$

Stochastic Gradient Descent (SGD)

Mini-Batch Gradient Descent
Instead of full gradient:

$$\nabla L(\boldsymbol{\beta}) = -\frac{2}{n}\mathbf{X}^\top \mathbf{r} \tag{13}$$

Use mini-batch of size m:

$$\nabla L_B(\boldsymbol{\beta}) = -\frac{2}{m}\mathbf{X}_B^\top \mathbf{r}_B \tag{14}$$

- $m = 1$: Stochastic GD (noisy but fast)
- $m = n$: Batch GD (stable but slow)
- $m \in [32, 256]$: Mini-batch (good tradeoff)

SGD: Process data once per epoch, update many times

R-Squared (R^2)

Coefficient of Determination

$$R^2 = 1 - \frac{\text{SS}_{\text{res}}}{\text{SS}_{\text{tot}}} = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \tag{15}$$

Interpretation:

- Proportion of variance explained by model
- $R^2 = 0$: Model no better than mean
- $R^2 = 1$: Perfect fit
- $R^2 = 0.7$: 70% of variance explained

R^2 **always increases with more features – use Adjusted** R^2

Adjusted R-Squared

Penalizing Model Complexity

$$R^2_{\text{adj}} = 1 - \frac{(1 - R^2)(n - 1)}{n - p - 1} \tag{16}$$

Properties:

- Adjusts for number of predictors p
- Can decrease when adding irrelevant features
- Better for model comparison

Use R^2_{adj} when comparing models with different p

RMSE and MAE

Error Metrics in Original Units

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2} \tag{17}$$

$$\text{MAE} = \frac{1}{n}\sum|y_i - \hat{y}_i| \tag{18}$$

Comparison:

- RMSE: Penalizes large errors more (sensitive to outliers)
- MAE: More robust, easier to interpret
- Units: Same as target variable (e.g., dollars)

Report both for comprehensive evaluation

Standard Errors of Coefficients

Quantifying Uncertainty in Estimates

$$\text{Var}(\hat{\boldsymbol{\beta}}) = \sigma^2(\mathbf{X}^\top\mathbf{X})^{-1} \tag{19}$$

Standard Error of $\hat{\beta}_j$:

$$\text{SE}(\hat{\beta}_j) = \hat{\sigma}\sqrt{[(\mathbf{X}^\top\mathbf{X})^{-1}]_{jj}} \tag{20}$$

where $\hat{\sigma}^2 = \frac{1}{n-p-1}\sum(y_i - \hat{y}_i)^2$ (unbiased variance estimate)

SE tells us how much $\hat{\beta}_j$ would vary across different samples

Hypothesis Testing for Coefficients

Is Feature j Significant?

- $H_0 : \beta_j = 0$ (feature has no effect)
- $H_1 : \beta_j \neq 0$ (feature matters)

t-Statistic:

$$t_j = \frac{\hat{\beta}_j - 0}{\mathrm{SE}(\hat{\beta}_j)} \sim t_{n-p-1} \tag{21}$$

Decision Rule:

- p-value < 0.05: Reject H_0, coefficient is significant
- p-value ≥ 0.05: Cannot reject H_0

Always check p-values before interpreting coefficients

Confidence Intervals

95% CI for Coefficient β_j:

$$\hat{\beta}_j \pm t_{n-p-1,0.975} \times \text{SE}(\hat{\beta}_j) \tag{22}$$

Interpretation: If we repeated the study many times, 95% of intervals would contain the true β_j

Prediction Intervals:

- **Confidence interval for mean:** $\hat{y} \pm t \cdot \text{SE}(\hat{y})$
- **Prediction interval for new observation:** Wider (includes σ^2)

CI for mean is narrower; prediction interval accounts for individual variability

F-Test for Overall Model Significance

Null hypothesis: $H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$ (model has no explanatory power)

$$F = \frac{R^2/p}{(1 - R^2)/(n - p - 1)} \sim F_{p,n-p-1} \tag{23}$$

- Tests whether the model **as a whole** explains significant variance
- Reject H_0 if $F > F_{\alpha,p,n-p-1}$ (or equivalently, if p-value $< \alpha$)
- Complements individual t-tests: possible to have no significant t-tests but a significant F-test (multicollinearity)

Equivalently (using sums of squares):

$$F = \frac{(\text{TSS} - \text{RSS})/p}{\text{RSS}/(n - p - 1)} = \frac{\text{MSR}}{\text{MSE}} \tag{24}$$

Every regression output table reports the F-statistic. Always check it before interpreting individual coefficients.

OLS and Maximum Likelihood Estimation

Under the normality assumption $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$, the likelihood is:

$$L(\boldsymbol{\beta}, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - \mathbf{x}_i'\boldsymbol{\beta})^2}{2\sigma^2}\right) \tag{25}$$

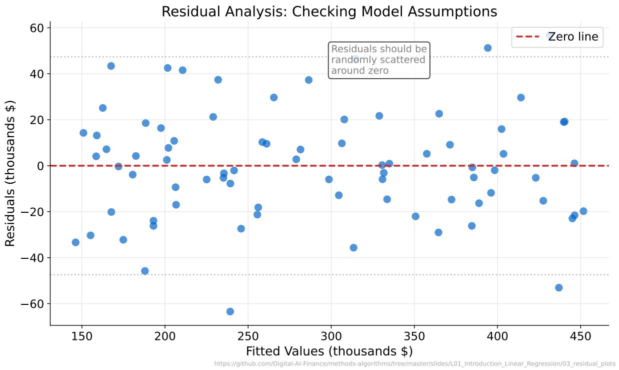
Maximizing the log-likelihood $\ell = -\frac{n}{2}\ln(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum(y_i - \mathbf{x}_i'\boldsymbol{\beta})^2$:

- $\frac{\partial \ell}{\partial \boldsymbol{\beta}} = 0$ gives the **same normal equations as OLS**
- $\hat{\sigma}^2_{MLE} = \frac{\text{RSS}}{n}$ (biased; OLS uses $n - p - 1$)
- This connection enables likelihood ratio tests, AIC/BIC model comparison

Key insight: OLS $\equiv$ MLE under normality $\Rightarrow$ all MLE properties (efficiency, asymptotic normality) transfer to OLS.

This bridges to L02 (Logistic Regression), where MLE is the only estimation method.

Residual Analysis



Good: random scatter. Bad: patterns indicate model misspecification

Formal Assumption Diagnostic Tests

Assumption	**Test**	H_0
Homoscedasticity	Breusch-Pagan	Constant variance
	White test	Constant variance (robust)
No autocorrelation	Durbin-Watson	No serial correlation
Normality	Shapiro-Wilk	Residuals are normal
	Jarque-Bera	Skewness=0, kurtosis=3
Functional form	Ramsey RESET	No omitted nonlinearities

When assumptions fail:

- Heteroscedasticity $\Rightarrow$ White/HC robust standard errors
- Autocorrelation $\Rightarrow$ Newey-West standard errors, GLS
- Non-normality $\Rightarrow$ Bootstrap inference (large n: CLT helps)

In finance/banking, regulators require formal test results – "the residuals looked fine" is insufficient.

Influence Diagnostics: Hat Matrix and Cook's Distance

The **hat matrix** maps observed to fitted values: $\hat{\mathbf{y}} = \mathbf{H}\mathbf{y}$ where

$$\mathbf{H} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}' \tag{26}$$

Leverage: $h_{ii} \in [1/n, 1]$ measures how far $\mathbf{x}_i$ is from the center of the design space. High leverage ($h_{ii} > 2p/n$): observation *could* be influential.
Cook's Distance combines leverage and residual:

$$D_i = \frac{e_i^2}{p \cdot \text{MSE}} \cdot \frac{h_{ii}}{(1 - h_{ii})^2} \tag{27}$$

- $D_i > 1$: observation substantially changes $\hat{\beta}$ when removed
- Critical in finance: a single crisis observation can distort the entire model

Rule of thumb: investigate points with $D_i > 4/n$ or leverage $h_{ii} > 2(p+1)/n$.

Train-Test Split

Evaluating Generalization

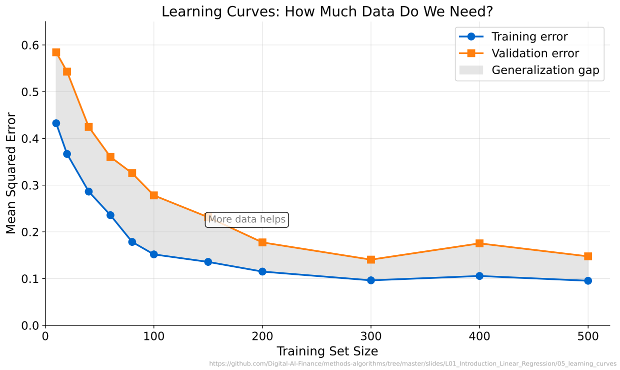
- Never evaluate on training data alone
- Split: 70-80% train, 20-30% test
- Report test set metrics

Cross-Validation (K-Fold):

- Split into K folds (typically $K = 5$ or 10)
- Train on $K - 1$ folds, validate on 1
- Repeat K times, average results

CV gives more reliable estimate with limited data

Learning Curves



Gap between curves indicates overfitting; convergence shows saturation

The Overfitting Problem

When Models Memorize Instead of Learn

- High-dimensional data ($p \approx n$ or $p > n$)
- Coefficients become very large
- Perfect fit on training data, poor generalization

Solution: Add Penalty to Loss Function

$$L_{\text{reg}}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda \cdot \text{penalty}(\boldsymbol{\beta}) \tag{28}$$

λ **controls strength of regularization**

Ridge Regression (L2)

L2 Penalty: Sum of Squared Coefficients

$$L_{\text{ridge}}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_2^2 \tag{29}$$

Closed-Form Solution:

$$\hat{\boldsymbol{\beta}}_{\text{ridge}} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y} \tag{30}$$

- Shrinks all coefficients toward zero
- Never sets coefficients exactly to zero
- Always invertible (even when $p > n$)

Ridge adds λ to diagonal – stabilizes inversion

Lasso Regression (L1)

L1 Penalty: Sum of Absolute Coefficients

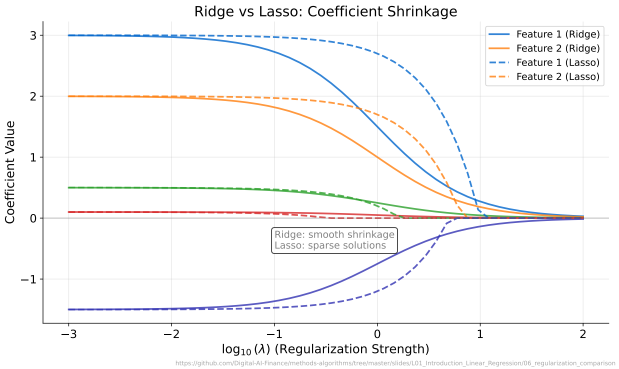
$$L_{\text{lasso}}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \lambda\|\boldsymbol{\beta}\|_1 \tag{31}$$

Properties:

- Produces sparse solutions (some $\beta_j = 0$)
- Automatic feature selection
- No closed-form solution (use coordinate descent)

Lasso: Least Absolute Shrinkage and Selection Operator

Ridge vs Lasso Comparison



Ridge: smooth shrinkage. Lasso: sparse (feature selection)

Elastic Net

Combining L1 and L2 Penalties (Zou & Hastie, 2005)

$$\min_{\beta} \frac{1}{2N}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2 + \lambda \left(\alpha\|\boldsymbol{\beta}\|_1 + \frac{1-\alpha}{2}\|\boldsymbol{\beta}\|_2^2 \right) \tag{32}$$

where $\alpha \in [0, 1]$ is the mixing parameter.

Benefits:

- Sparsity from L1 when $\alpha > 0$
- Stability from L2 (handles correlated features)
- $\alpha = 1$: pure Lasso; $\alpha = 0$: pure Ridge

Often best of both worlds for correlated features

Choosing Lambda

Cross-Validation for Hyperparameter Tuning

1. Define grid of λ values (e.g., 10^{-4} to 10^{4})
2. For each λ, perform K-fold CV
3. Select λ with lowest CV error
4. Refit on full training data

In Practice:

- `sklearn.linear_model.RidgeCV`
- `sklearn.linear_model.LassoCV`

Larger λ = more regularization = simpler model

Decomposing Prediction Error

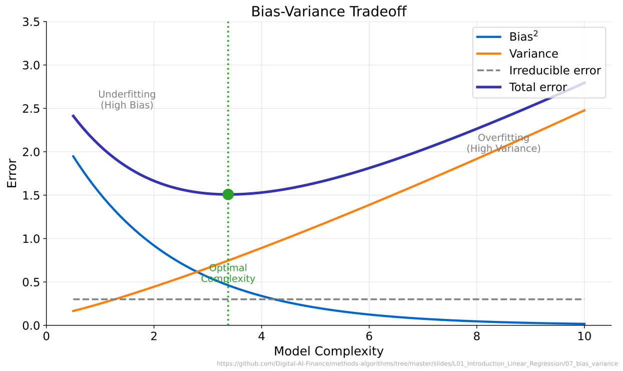
Expected Prediction Error

$$E[(y - \hat{f}(x))^2] = \text{Bias}^2(\hat{f}) + \text{Var}(\hat{f}) + \sigma^2 \tag{33}$$

- **Bias**: Error from wrong assumptions (underfitting)
- **Variance**: Error from sensitivity to training data (overfitting)
- σ^2: Irreducible noise in data

We can't reduce irreducible error – focus on bias and variance

The Tradeoff Illustrated



Optimal complexity minimizes total error

Regularization and Bias-Variance

How Regularization Helps

- Increasing λ: **increases bias**, **decreases variance**
- Decreasing λ: decreases bias, increases variance
- Optimal λ: minimizes total error

In Practice:

- Use CV to find optimal λ
- Regularization almost always helps when p is large

Regularization trades a little bias for a lot of variance reduction

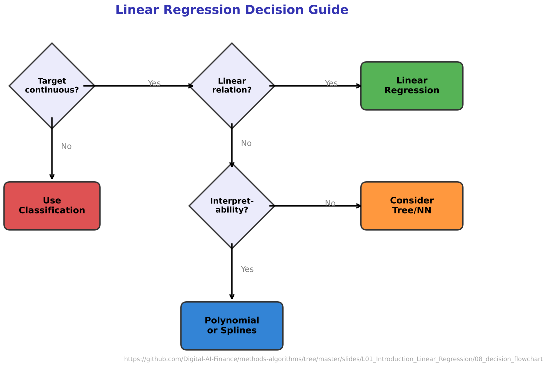
Hands-on Exercise

Open the Colab Notebook

- Exercise 1: Implement OLS from scratch
- Exercise 2: Use scikit-learn LinearRegression
- Exercise 3: Compare with gradient descent

Link: See course materials on GitHub

Algorithm Selection Guide



Linear Regression Decision Guide
Target continuous?
Yes
No
Use Classification
Linear relation?
Yes
Linear Regression
No
Interpret-ability?
No
Consider Tree/NN
Yes
Polynomial or Splines
https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L01_Introduction_Linear_Regression/08_decision_flowchart

Use this framework when choosing regression methods

Linear Regression: When and Why

Use When:

- Continuous target variable
- Approximate linear relationships
- Interpretability is critical
- Inference on coefficients needed
- Fast prediction required

Avoid When:

- Target is categorical
- Strong non-linear patterns
- Many outliers present
- Features highly correlated
- Prediction accuracy paramount

When in doubt, linear regression is a strong baseline

Diagnosing Multicollinearity

Variance Inflation Factor (VIF)

$$\text{VIF}_j = \frac{1}{1 - R_j^2} \tag{34}$$

where R_j^2 is from regressing x_j on all other features.
Interpretation:

- $\text{VIF} = 1$: No correlation with other features
- $\text{VIF} > 5$: Moderate concern
- $\text{VIF} > 10$: Serious multicollinearity

Remedies:

- Remove highly correlated features
- Use Ridge regression ($\lambda > 0$ stabilizes)
- Apply PCA before regression

Always check VIF before trusting coefficient estimates

Key Equations Summary

$$\text{Model:} \quad \mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{35}$$

$$\text{OLS Solution:} \quad \hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} \tag{36}$$

$$\text{Gradient:} \quad \nabla L = -2\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \tag{37}$$

$$\text{GD Update:} \quad \boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \alpha\nabla L \tag{38}$$

$$\text{Ridge:} \quad \hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y} \tag{39}$$

$$R^2: \quad 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \tag{40}$$

Key Takeaways

1. Linear regression minimizes squared error – closed form or GD
2. Matrix notation enables efficient computation
3. Gradient descent scales to large datasets
4. Regularization (Ridge/Lasso) prevents overfitting
5. The bias-variance tradeoff guides model complexity
6. Always evaluate on held-out test data

Next Session: Logistic Regression for Classification

References

- James, Witten, Hastie, Tibshirani (2021). *Introduction to Statistical Learning*. Chapter 3.
- Hastie, Tibshirani, Friedman (2009). *Elements of Statistical Learning*. Chapter 3.
- Bishop (2006). *Pattern Recognition and Machine Learning*. Chapter 3.

Online Resources:

- scikit-learn: `https://scikit-learn.org/stable/modules/linear_model.html`
- Stanford CS229: `https://cs229.stanford.edu/`