Introduction & Linear Regression

Overview

Methods and Algorithms

MSc Data Science

Spring 2026

Outline

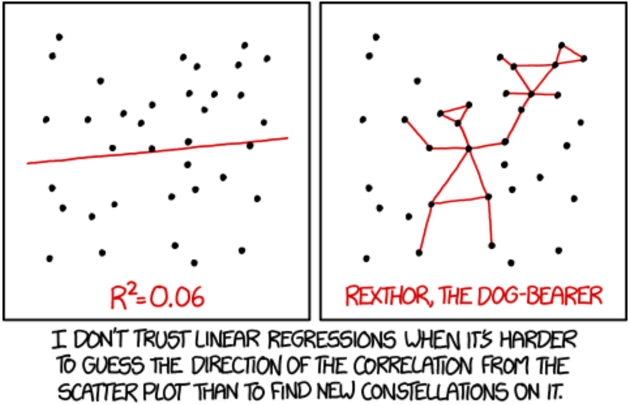
Learning Objectives

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

1. **Understand** the ordinary least squares (OLS) framework
2. **Apply** gradient descent for parameter optimization
3. **Interpret** regression coefficients in business contexts

Finance Applications: Property valuation, asset pricing (CAPM)

Foundation for all supervised learning methods



XKCD #1725 by Randall Munroe (CC BY-NC 2.5) – Always check if your data supports a linear fit!

The Business Problem

Finance Use Case: Predicting House Prices

- Banks need accurate property valuations for mortgages
- Insurance companies assess property risk
- Investors evaluate real estate portfolios

The Question: Given features (size, location, age), what price?

Why Linear Regression?

- Interpretable coefficients (price per square meter)
- Fast, well-understood method
- Strong baseline for comparison

Linear regression: the "hello world" of machine learning

What is Linear Regression?

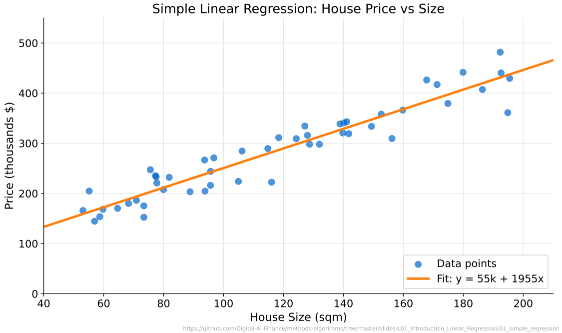
Core Idea: Model the relationship between inputs and a continuous output

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \varepsilon \tag{1}$$

- y: Target variable (house price)
- $x_1, \ldots, x_p$: Features (size, bedrooms, age)
- $\beta_0, \ldots, \beta_p$: Coefficients to learn
- ε: Random error term

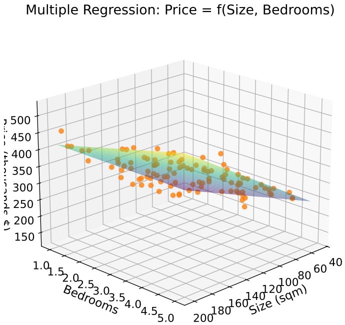
Goal: Find coefficients that minimize prediction error

Simple Linear Regression



The best-fit line minimizes the sum of squared distances from points

Multiple Linear Regression



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

With multiple features, we fit a hyperplane to minimize squared errors

The Optimization Problem

Objective: Minimize the sum of squared errors (SSE)

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

Why Squared Errors?

- Penalizes large errors more than small errors
- Differentiable (enables gradient-based optimization)
- Leads to closed-form solution

This objective function defines "ordinary least squares" (OLS)

1. Closed-Form (Normal Equation)

$$\hat{\beta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} \tag{3}$$

Pros:

- Exact solution
- One computation

Cons:

- Slow for large p
- Memory intensive

2. Gradient Descent

$$\beta_{t+1} = \beta_t - \alpha \nabla L(\beta_t) \tag{4}$$

Pros:

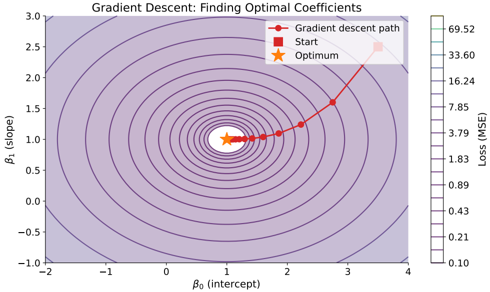
- Scales to big data
- Memory efficient

Cons:

- Requires tuning α
- Iterative process

Both methods yield the same solution for OLS

Gradient Descent in Action



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L01_Introduction_Linear_Regression/04_gradient_descent

Iteratively update parameters in the direction of steepest descent

Interpreting Coefficients

Example: House price model

$$\text{Price} = 50,000 + 200 \times \text{Size} + 15,000 \times \text{Bedrooms} - 1,000 \times \text{Age} \tag{5}$$

Interpretation:

- $\beta_0 = 50,000$: Base price (all features $= 0$)
- $\beta_1 = 200$: Each extra sqm adds \$200
- $\beta_2 = 15,000$: Each bedroom adds \$15,000
- $\beta_3 = -1,000$: Each year of age subtracts \$1,000

Coefficients show marginal effect, holding others constant

Model Evaluation

Key Metrics:

- R^2 **(Coefficient of Determination):** Variance explained

$$R^2 = 1 - \frac{\sum (y_i - \hat{y}_i)^2}{\sum (y_i - \bar{y})^2} \tag{6}$$

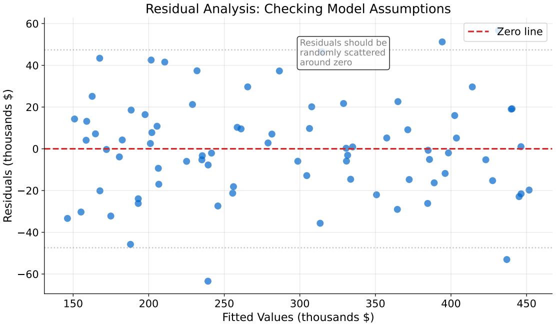
- **RMSE (Root Mean Squared Error):** Prediction accuracy

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum (y_i - \hat{y}_i)^2} \tag{7}$$

In Practice: $R^2 = 0.75$ means model explains 75% of price variance

Always evaluate on held-out test data

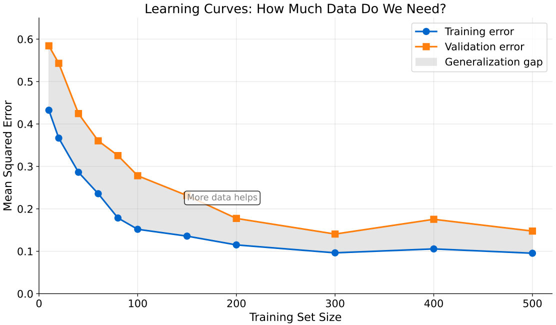
Residual Diagnostics



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L01_Introduction_Linear_Regression/03_residual_plots

Good residuals are random; patterns suggest model problems

Learning Curves



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L01_Introduction_Linear_Regression/05_learning_curves

Learning curves help diagnose underfitting vs overfitting

When to Use Linear Regression

Use When:

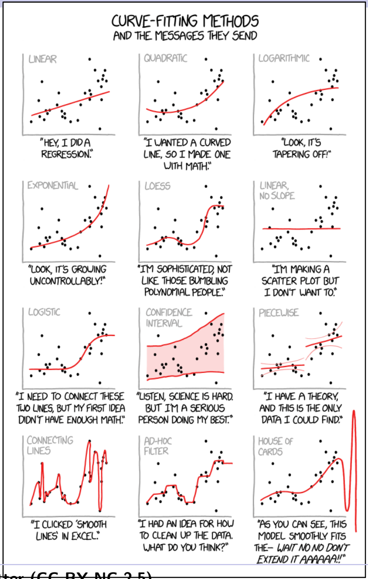
- Target is continuous
- Linear relationships expected
- Interpretability matters
- Fast inference needed

Avoid When:

- Target is categorical
- Strong non-linearities
- High multicollinearity
- Many outliers present

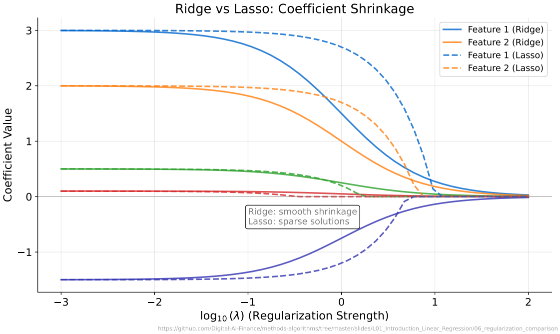
Pros	**Cons**
Interpretable	Assumes linearity
Fast training	Sensitive to outliers
Well-understood	Limited flexibility

The Danger of Overfitting



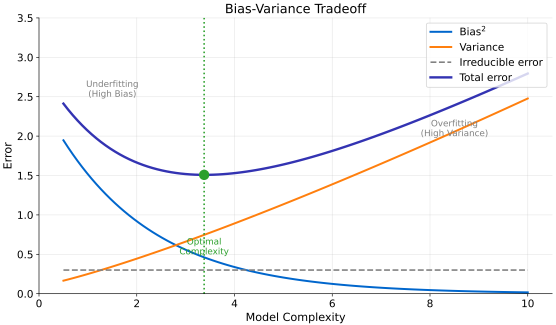
XKCD #2048 – Simpler models often generalize better (CC BY-NC 2.5)

Regularization: Ridge vs Lasso



Ridge shrinks all coefficients; Lasso can set some to exactly zero

Bias-Variance Tradeoff



https://github.com/Digital-AI-Finance/methods-algorithms/tree/master/slides/L01_Introduction_Linear_Regression/07_bias_variance

Model complexity controls the tradeoff between bias and variance

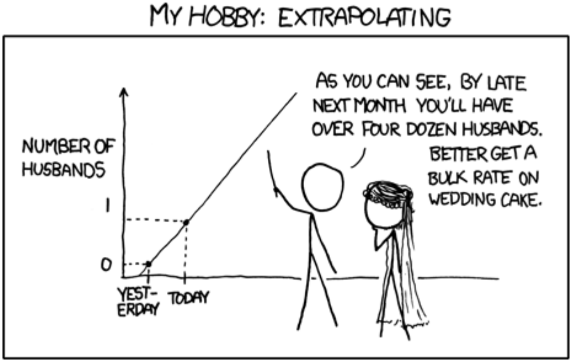
Hands-on Exercise

Open the Colab Notebook

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

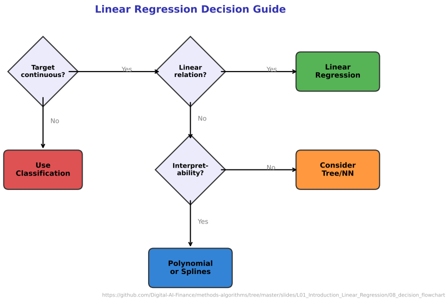
Link: See course materials on GitHub

A Word of Caution



XKCD #605 by Randall Munroe (CC BY-NC 2.5) – Extrapolation is dangerous!

Decision Framework



Use this flowchart to decide when linear regression is appropriate

Key Takeaways

1. Linear regression models continuous outcomes as weighted sum of features
2. OLS minimizes sum of squared errors
3. Solve via normal equation (small data) or gradient descent (big data)
4. Coefficients are directly interpretable as marginal effects
5. Evaluate using R^2 (variance explained) and RMSE (prediction error)

Next: Deep dive into mathematics and implementation

References:

- ISLR Chapter 3: Linear Regression
- ESL Chapter 3: Linear Methods for Regression