# Introduction & Linear Regression

## Overview

Methods and Algorithms

MSc Data Science

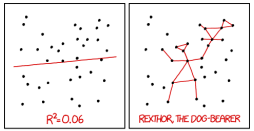
Spring 2026

# Outline

# What is Machine Learning?

**The Big Idea**

- Teaching computers to learn from data, without being explicitly programmed
- Everyday examples: email spam filters, Netflix recommendations, voice assistants
- Key idea: the computer finds patterns in data, then uses those patterns to make predictions on new data




**XKCD #1725 by Randall Munroe (CC BY-NC 2.5) – ML is already part of your daily life**

# Supervised vs Unsupervised Learning

**Supervised Learning**

- We have the answers (labels) for training data
- Goal: predict the answer for new data
- Examples: predicting house prices, classifying emails

**Unsupervised Learning**

- No answers – just data
- Goal: find hidden structure or groups
- Examples: customer segments, anomaly detection

**This course covers both: L01–L04 supervised, L05 unsupervised**

# What is Regression?

**Predicting a Number (Not a Category)**

- Predicting a number (not a category) – for example, a house price
- Example: predicting house price from its size in square meters
- We draw the best line through the data points
- The line captures the relationship between input and output

**Regression = predicting a continuous value. Classification = predicting a category.**

# The Simplest Model

**The Idea**

Price goes up as size increases. We write this as:

$$y = a + b \times x$$

- $y$ = price (what we predict)
- $x$ = size (what we know)
- $a$ = starting price, $b$ = price per sqm

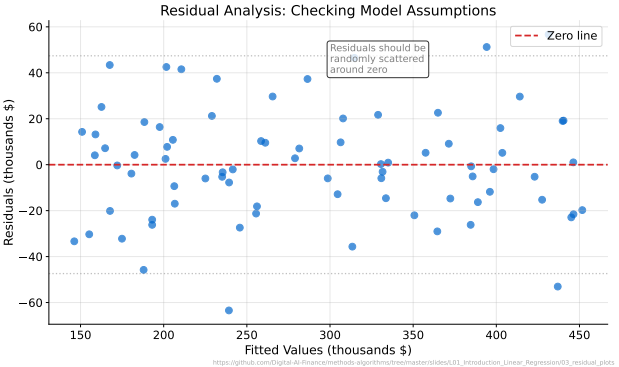
**Example**

$$\text{Price} = 50{,}000 + 200 \times \text{Size}$$

- A 100 sqm apartment: $50{,}000 + 200 \times 100 = 70{,}000$
- A 150 sqm apartment: $50{,}000 + 200 \times 150 = 80{,}000$
- Each extra square meter adds 200 to the price

**This is the entire idea of linear regression – the rest is details**

# What Makes a Good Prediction?




- The red lines show prediction errors (residuals)
- A good model has small, random errors
- Patterns in errors = something the model missed

**We want errors to be small and random – not systematic**

# How Does the Computer Learn?

**The Learning Process**

**Step 1:** Start with a random line

**Step 2:** Measure how wrong it is (errors)

**Step 3:** Adjust the line to reduce errors

**Step 4:** Repeat until errors are small

**Step 5:** Use the final line to predict

This process is called **optimization** – the computer tries thousands of lines and keeps the best one

**The computer tries thousands of lines and keeps the best one**

# Key Vocabulary

**Data**

- Features (inputs): size, bedrooms, age
- Target (output): price
- Observation: one data point (one house)

**Learning**

- Training: fitting the model to data
- Coefficients: the numbers the model learns
- Loss: how wrong the model is

**Evaluation**

- Prediction: model output for new data
- Error: difference between prediction and truth
- Overfitting: memorizing instead of learning

**Master these terms – they appear in every ML method we study**

# Road Map for Today

Now that you know the basics, we go deeper:

1. A real business problem (house prices for banks)
2. The math behind finding the best line
3. How to tell if our model is good
4. How to prevent our model from being too complex

**Everything ahead builds on the intuition from this intro**

# The Business Problem

**Finance Use Case**

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

**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**

# Linear Regression Formula

**The Model**

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon \quad (1)$$

- $y$: target variable (house price)
- $x_1, \ldots, x_p$: features (size, bedrooms, age)
- $\beta_0, \ldots, \beta_p$: coefficients to learn
- $\varepsilon$: random error term

**Worked Example**

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

- Base price: \$50,000
- Each sqm adds \$200
- Each bedroom adds \$15,000
- Each year of age subtracts \$1,000

**Goal: Find coefficients that minimize prediction error**

# The Optimization Problem

**Objective:** Minimize the sum of squared errors (SSE)

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

**Why Squared Errors?**

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

---

**This defines "ordinary least squares" (OLS)**

# Normal Equation vs Gradient Descent

**Normal Equation**

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

- Exact, closed-form solution
- One computation – no iterations
- Slow for very large datasets

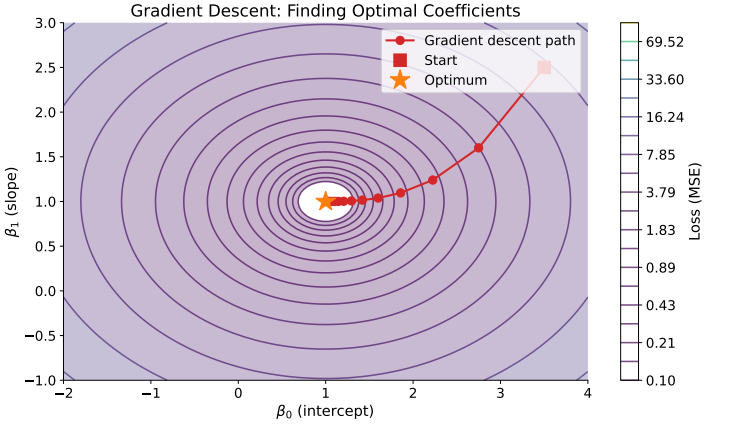
**Gradient Descent**

$$\boldsymbol{\beta}_{t+1} = \boldsymbol{\beta}_t - \alpha \nabla L(\boldsymbol{\beta}_t) \tag{4}$$

- Scales to big data
- Memory efficient
- Requires tuning learning rate $\alpha$

**Both methods yield the same solution for OLS**

# Gradient Descent in Action




**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

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

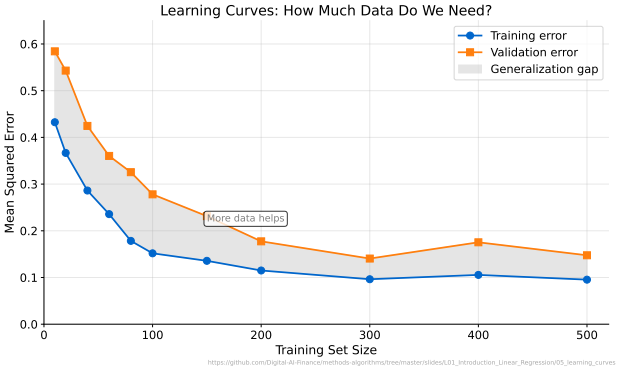
- **RMSE (Root Mean Squared Error):** Prediction accuracy

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

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

**Always evaluate on held-out test data**

# Learning Curves




- Gap between train and test error = overfitting
- Curves converging = more data won't help

**Learning curves diagnose underfitting vs overfitting**

# When to Use Linear Regression

**Advantages**

- \+ Interpretable coefficients
- \+ Fast training and prediction
- \+ Well-understood theory
- \+ Works as strong baseline

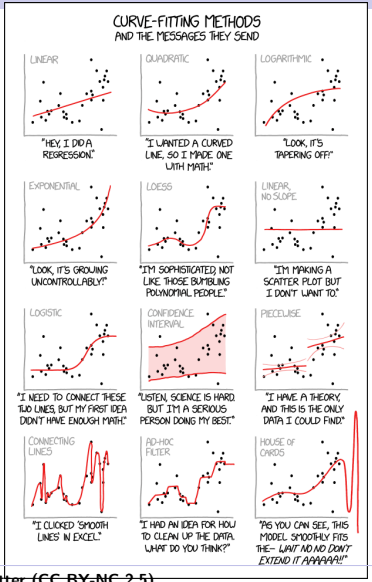
**Disadvantages**

- – Assumes linear relationships
- – Sensitive to outliers
- – Limited flexibility
- – Needs feature engineering for non-linear patterns

---

**When in doubt, start with linear regression as your baseline**

# The Danger of Overfitting




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

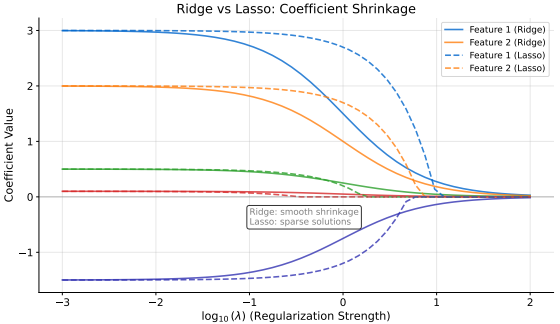
# Regularization Overview

**Ridge (L2)**

- Adds penalty on squared coefficients
- Shrinks all coefficients toward zero
- Never removes features entirely

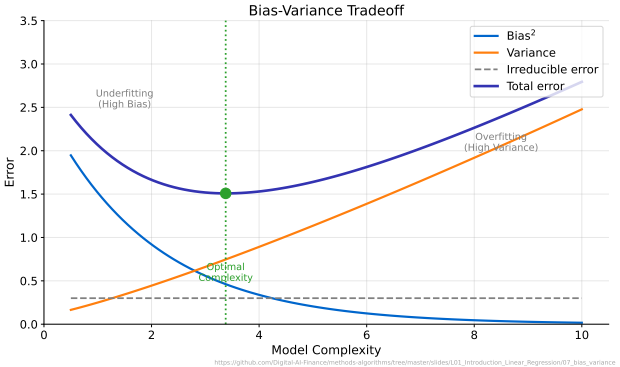
**Lasso (L1)**

- Adds penalty on absolute coefficients
- Can set coefficients exactly to zero
- Automatic feature selection




**Regularization prevents overfitting by penalizing large coefficients**

# Bias-Variance Tradeoff




- Bias: error from wrong assumptions (too simple)
- Variance: error from sensitivity to training data (too complex)
- Sweet spot: minimize total error

**Model complexity controls the tradeoff between bias and variance**

# Hands-on Exercise

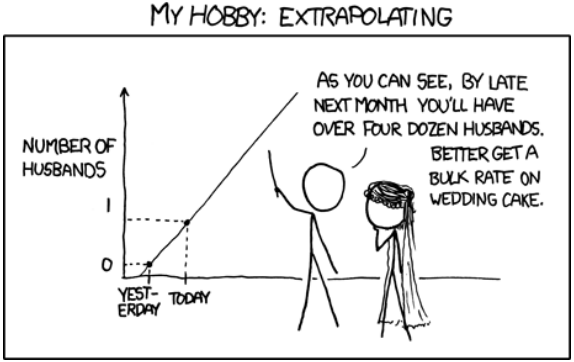
**Open the Colab Notebook**

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

**Link:** See course materials on GitHub

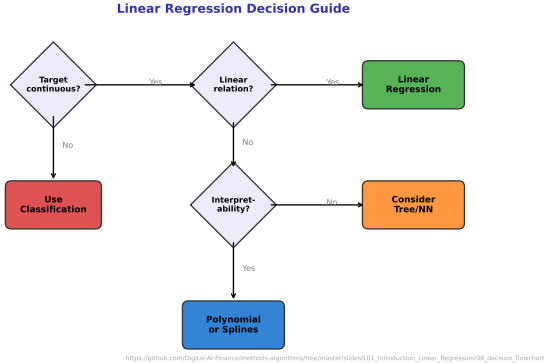
**Start with Exercise 2 if short on time**

# 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

**Key Concepts**

- Linear regression predicts continuous outcomes
- OLS minimizes sum of squared errors
- Solve via normal equation or gradient descent

**Next:** Deep dive into mathematics and implementation

**References:**

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

**Practical Guidance**

- Coefficients are directly interpretable
- Evaluate with $R^2$ and RMSE on test data
- Use regularization to prevent overfitting

**Foundation for all supervised learning methods**