2.1 - Training, Validation and Testing

Giovanni Della Lunga
giovanni.dellalunga@unibo.it

Introduction to Machine Learning for Finance

Bologna - February-March, 2025

Why This Lecture Matters

Even before discussing specific machine learning models, we must define a rigorous framework for:

- training a model on observed data,
- selecting among alternative specifications,
- evaluating generalization on unseen data.

In finance, this step is not a technicality: without a correct evaluation protocol, apparent "predictive power" often disappears out-of-sample.

Educational Goals

At the end of this lecture, you should be able to:

- Clearly distinguish **training**, **validation**, and **test** data and explain why each is needed.
- Explain the difference between **model selection** and **model assessment**.
- Describe how **overfitting** emerges from iterative trial-and-error on the same dataset.
- Understand why the **test set must be treated as a scientific control**.
- Recognize why **time ordering** is a causal constraint in financial data.

Introduction

Introduction

- Before delving into specific machine learning models, it is crucial to establish a rigorous understanding of how to **train**, **validate** and **test** models properly.
- This step is not merely a technical detail but a **fundamental prerequisite** for ensuring that any model we build is reliable, interpretable, and capable of generalizing to new data.
- Without a robust evaluation framework, even the most sophisticated models can yield misleading results, leading to overconfidence in predictions that do not hold up in real-world applications.

Introduction

- In the next slides we're going to introduce several key concepts that are **transversal to all of machine learning**.
- Among these, the notions of **overfitting** and **bias and variance** play a central role.
- In particular bias and variance, often in tension with each other, govern the fundamental trade-offs in model performance:
  - a model with high bias oversimplifies the problem and fails to capture essential patterns...
  - whereas a model with high variance is overly sensitive to training data and performs poorly on unseen examples
- Understanding how to balance these factors is **critical for designing robust machine learning systems**.

Introduction

- To develop intuition for these ideas, we will use a **simple regression model** as our primary tool.
- This choice is intentional: rather than getting caught up in the intricacies of a complex algorithm, we will focus on the core issues of **model training and validation, bias-variance tradeoff, and generalization**.
- Regression models provide a clear and interpretable framework for illustrating these concepts, making them an ideal starting point before extending these ideas to more advanced models.

Introduction

In the following sections, we will cover the essential components of model validation, including:

- The distinction between training, validation, and test sets
- Bias-variance decomposition and its implications
- Overfitting and underfitting: causes and remedies
- Regularization Methods

By mastering these principles early, we establish a solid foundation that will allow us to critically assess the performance of any machine learning model we encounter. With these concepts in place, we can confidently proceed to more advanced methods, knowing that we have the tools to rigorously evaluate their effectiveness.

Train, Test and Validation Dataset

Model Selection vs Model Assessment

In machine learning, two goals are conceptually distinct:

- **Model selection**: choosing among alternative models or hyperparameter configurations.
- **Model assessment**: estimating the true out-of-sample performance of the final chosen model.

These two goals require **different data**.

Key principle:

- The **validation set** is used to *choose*.
- The **test set** is used to *judge*.

Using the same data to choose and to judge leads to overly optimistic conclusions.

Why Do We Need a Validation Set?

The validation set plays a specific and irreplaceable role:

- It allows comparison between models of different complexity.
- It guides hyperparameter tuning.
- It helps detect overfitting before deployment.

The validation set must be:

- out-of-sample relative to training data,
- **not** used for final performance claims.

Important warning: repeatedly "peeking" at test data turns it into training data.

The Test Set as a Scientific Control

The test set should be used **once**, at the end of the modeling process.

Think of it as a **control experiment**:

- It provides an unbiased estimate of performance on unseen data.
- It protects against unconscious over-optimization.
- It prevents "researcher degrees of freedom" from contaminating evaluation.

Rule: any decision influenced by test performance invalidates the test.

If the test set influences decisions, you no longer have a true out-of-sample evaluation.

The “Leakage” Mechanism (Conceptual View)

Information leakage is any situation where the training process uses information that would not be available at prediction time.

Leakage can occur even without explicit cheating:

- tuning choices informed by test results,
- preprocessing fit on the full dataset,
- selecting features using future data,
- selecting assets that survive to the end of the sample.

Key idea: leakage often appears as “excellent backtest performance” that collapses out-of-sample.

Time Ordering Is a Causal Constraint (Finance)

For time-dependent data, random shuffling breaks the causal structure.

In finance, the information set at time t must only include what is available up to t.

Therefore:

- training must precede validation and test **in time**,
- preprocessing must be fit on past data only,
- evaluation must mimic the real forecasting/decision environment.

This principle will guide all future model evaluation protocols in the course.

Example: Training and Validation

- When data is used for forecasting there is a danger that the machine learning model **will work very well for training data, but will not generalize well to other data**;
- An obvious point is that it is important that the data used in a machine learning model be representative of the situations to which the model is to be applied;
- It is also important to test a model **out-of-sample**, by this we mean that the model should be tested on data that is different from the sample data used to determine the parameters of the model;
- As we have said, data scientist refer to the sample data as the **TRAINING set** and the data used to determine the accuracy of the model as the **VALIDATION set**;
- Often a **TEST set** is used as well as we explain later;

Example: Training and Validation

- We will illustrate the use of a training set and the validation data set with a very simple Example (Hull, Chapter 1);
- We suppose that awe are interested in forecasting the salaries of people from their age;
- This simple regression model is an example of supervised learning...

Table 1. Salary as a function of Age for a certain profession in a certain area)

Age (years)	Salary ($)
25	135,000
55	260,000
27	105,000
35	220,000
60	240,000
65	265,000
45	270,000
40	300,000
50	265,000
30	105,000

Figure 1. Scatter plot of Salary as a function of Age (see Table 1)

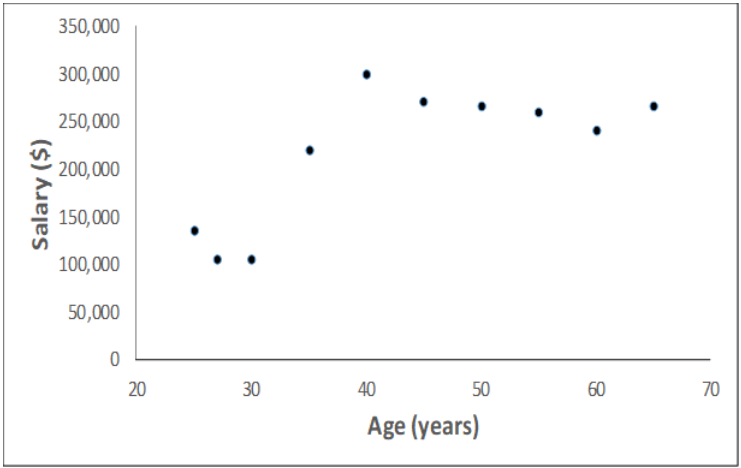
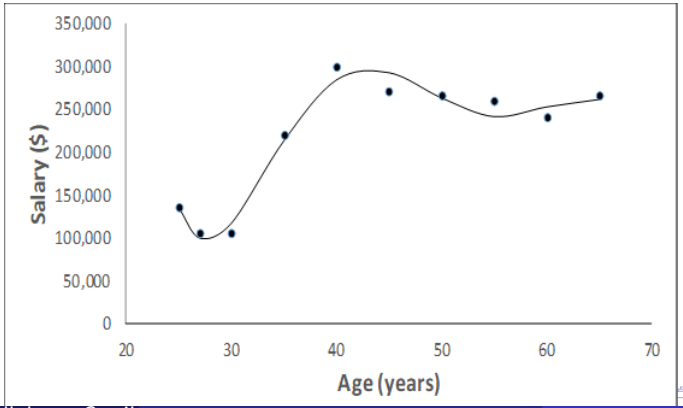


Figure 2. It's tempting to choose a model that fits the data really well for example with a polynomial of degree five (Y = Salary, X = Age):

$$Y = a + b_1X + b_2X^2 + b_3X^3 + b_4X^4 + b_5X^5$$



Discussion of Training Result

Example: Training and Validation

- The model provides a good fit to the data;
- The standard deviation of the difference between the salary given by the model and the actual salary for the ten individuals in the training data set (which is referred to as the **root mean square error (rmse)**) is $12902;
- However common sense would suggest that we many over-fitted the data;
- We need to check the model out-of-sample;
- To use the language of *data science* we need to determine whether the model generalizes well to a new data set that is different from the training set.

Figure 3. An Out-of-Sample Validation Set

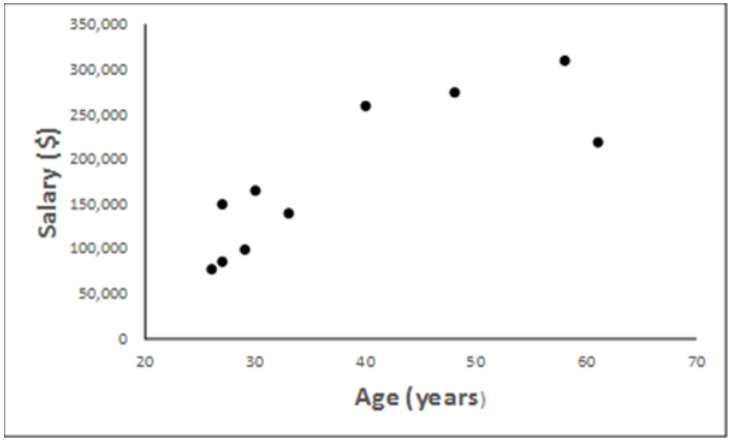
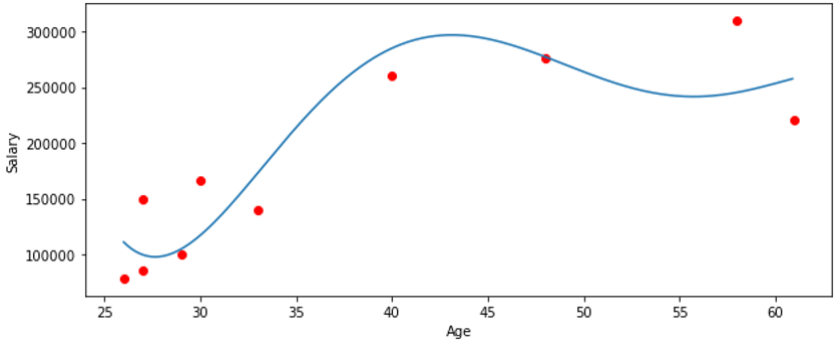


Figure 3. Scatter Plot for Validation Set



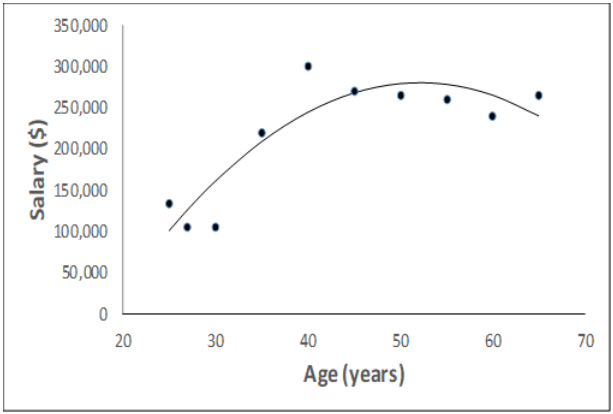
The Fifth Order Polynomial Model Does Not Generalize Well

- The root mean squared error (rmse) for the training data set is $12,902
- The rmse for the test data set is $38,794
- We conclude that the model overfits the data

Example: Training and Validation

Figure 4. A Simpler Quadratic Model

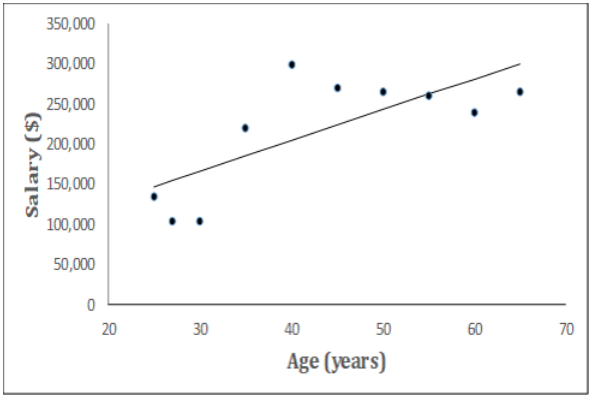
$$Y = a + b_1 X + b_2 X^2$$



Example: Training and Validation

Figure 5 . Linear Model

$$Y = a + b_1 X$$



Example: Training and Validation

Table 3. Summary of Results: The linear model under-fits while the 5th degree polynomial over-fits:

	Polynomial of degree 5	Quadratic model	Linear model
Training set	12, 902	32,932	49,731
Validation set	38,794	33,554	49,990

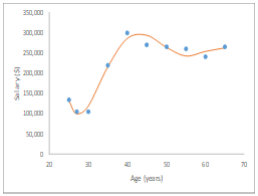
Example: Training and Validation

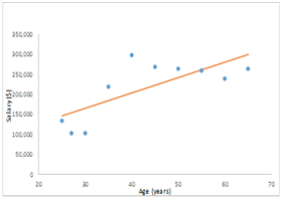
Table 3. Summary of Results: The linear model under-fits while the 5th degree polynomial over-fits:

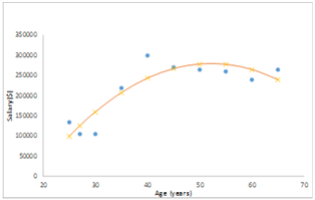
	Polynomial of degree 5	Quadratic model	Linear model
Training set	12,902	32,932	49,731
Validation set	38,794	33,554	49,990

Example: Training and Validation

Figure 6. Overfitting/Underfitting Example: predicting salaries for people in a certain profession in a certain area (only 10 observations)



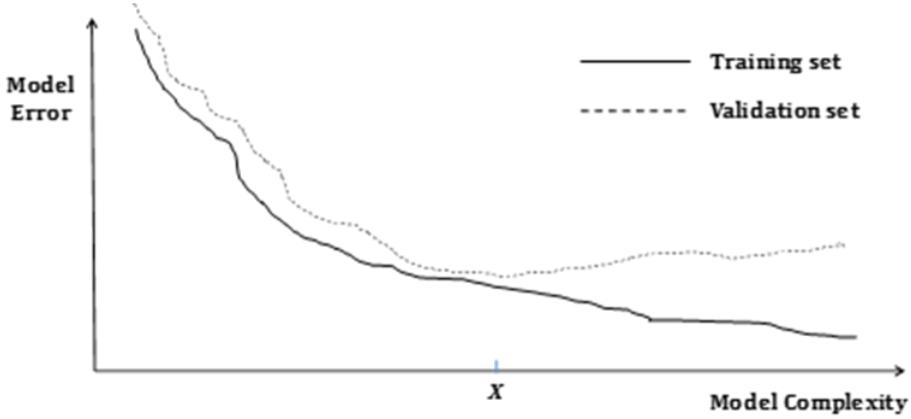




Overfitting ———— Underfitting ———— Best model?

Example: Training and Validation

Typical Pattern of Errors for Training Set and Validation Set



Example: Training and Validation

ML Good Practice

- Divide data into three sets
- Training set
- Validation set
- Test set
- Develop different models using the **training set** and compare them using the **validation set**;
- Rule of thumb: increase model complexity until model no longer generalizes well to the validation set;
- The **test set** is used to provide a final out-of-sample indication of how well the chosen model works;

A Useful Example . . . and Why It Can Mislead in Finance

The salary–age example is excellent to build intuition about **overfitting**:

- it clearly separates **training fit** from **out-of-sample performance**;
- it shows how model complexity can over-adapt to limited data.

However, finance differs in a crucial way:

- the target relationship is rarely stable: the data-generating process changes over time;
- the main source of error is often **structural uncertainty** (non-stationarity), not only i.i.d. noise.

Takeaway: in finance, the key question is not only *"How complex should my model be?"* but also *"Will the relationship still exist tomorrow?"*

From Noise to Structural Change: What Dominates in Finance

In a textbook regression setting:

$$y = f(x) + \varepsilon, \qquad \mathbb{E}[\varepsilon] = 0,\ \mathrm{Var}(\varepsilon) = \sigma^2$$

we often assume that f **is stable** and uncertainty is mostly in ε.

In finance, a more realistic view is:

$$y_t = f_t(x_t) + \varepsilon_t$$

where f_t **can drift** (regimes, policy changes, microstructure effects, market ecology).

Practical implication:

- even a low-variance estimator of f_t can fail if $f_{t+1} \neq f_t$;
- evaluation protocols must respect time ordering and aim to detect **instability**.

RMSE Is Not the Same as Economic Usefulness

In the salary–age example, RMSE is a meaningful objective: it measures prediction accuracy.

In finance, a lower prediction error does **not** automatically imply value:

- profits depend on **sign**, **timing**, **transaction costs**, and **risk**;
- small statistical improvements may be economically irrelevant;
- optimizing a symmetric loss can be misaligned with asymmetric payoffs and constraints.

Takeaway: always distinguish **statistical performance** (loss) from **economic performance** (utility, P&L, risk-adjusted metrics).

Bias and Variance

Bias and Variance

- Suppose there is a relationship between an independent variable x and a dependent variable y:

$$y = f(x) + \epsilon \tag{1}$$

  where ϵ is an error term with mean zero and variance σ^2.
- The error term captures either genuine randomness in the data or noise due to measurement error.
- Suppose we find, with a Machine Learning technique, a deterministic model for this relationship:

$$y = \hat{f}(x) \tag{2}$$

Bias and Variance

- Now it comes a new data point x' not in the training set and we want to predict the corresponding y';
- The error we will observe in our model at point x' is going to be

$$\hat{f}(x') - f(x') - \epsilon \tag{3}$$

- There are two different sources of error in this equation.
- The first one is included in the factor ϵ;
- The second one, more interesting, is due to what is in our training set.
- A robust model should give us the same prediction whatever data we used for training out model.

Bias and Variance

- Let's look at the average error:

$$E\left[\hat{f}(x')\right] - f(x') \tag{4}$$

where the expectation is taken over random samples of training data (having the same distributio as the training data).

- This is the definition of the **bias**

$$\text{Bias}\left[\hat{f}(x')\right] = E\left[\hat{f}(x')\right] - f(x') \tag{5}$$

The expectation is taken over all possible training sets drawn from the same data-generating process.

Bias and Variance

- We can also look at the mean square error

$$E\left[\left(\hat{f}(x') - f(x') - \epsilon\right)^2\right] = \left[\text{Bias}\left(\hat{f}(x')\right)\right]^2 + \text{Var}\left[\hat{f}(x')\right] + \sigma^2$$

  where we remember that $\hat{f}(x')$ and ϵ are independent.
- This show us that there are two important quantities, the **bias** and the **variance** that will affect our results and that we can control to some extent.

Bias and Variance

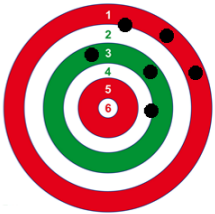
- **What is Bias?** It's the difference between the average prediction of our model and the correct value which we are trying to predict. Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.
- **What is Variance?** It's the variability of model prediction for a given data point or a value which tells us spread of our data. Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but has high error rates on test data.

Bias and Variance

Low Bias and Low Variance

High Bias and Low Variance

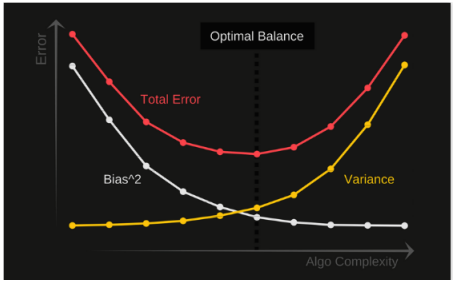
Low Bias and High Variance



High Bias and High Variance

Bias and Variance

Unfortunately, we often find that there is a trade-off between bias and variance. As one is reduced, the other is increased. This is the matter of over- and under-fitting.



Cross Validation and Model Selection

Why Cross-Validation Exists

A single train/validation split is often **high-variance**:

- the chosen split may be "lucky" or "unlucky";
- performance estimates may depend strongly on which observations end up in validation.

Cross-validation (CV) is a resampling scheme designed to:

- obtain a **more stable estimate** of out-of-sample performance;
- support **model selection** (algorithm choice and hyperparameters);
- reduce sensitivity to a particular split.

Core idea: reuse the data efficiently while keeping training and validation *separate* in each fold.

What CV Estimates (Conceptually)

Let $R(\mathcal{A})$ be the **true generalization risk** of a learning procedure $\mathcal{A}$ (e.g. expected loss on new data).

We cannot observe $R(\mathcal{A})$ directly. CV provides an **estimator**:

$$\widehat{R}_{CV}(\mathcal{A}) \;=\; \frac{1}{K}\sum_{k=1}^{K} \widehat{R}^{(k)},$$

where $\widehat{R}^{(k)}$ is the validation loss on fold k after training on the other $K-1$ folds.

Interpretation:

- $\widehat{R}_{CV}$ approximates expected out-of-sample error;
- the dispersion across folds provides a crude sense of **uncertainty** / stability.

K-Fold Cross-Validation: Procedure

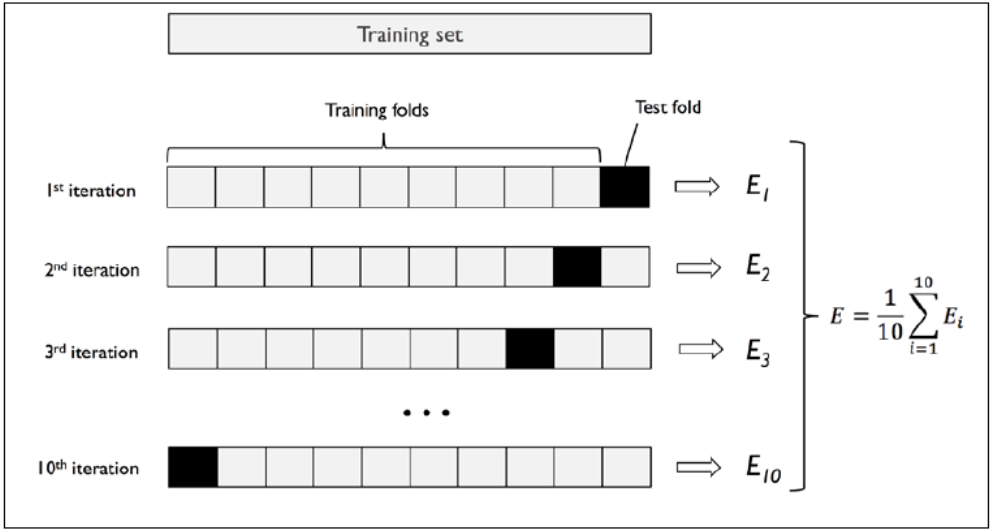
K-fold CV:

1. Split the dataset into K disjoint folds of (approximately) equal size.
2. For each $k = 1, \ldots, K$:
   - Train on $\{1, \ldots, K\} \setminus \{k\}$ folds;
   - Validate on fold k and record the score/loss.
3. Aggregate the K validation results (mean, and optionally std).

Output:

- an estimate of performance for a *fixed* modeling pipeline;
- a criterion to compare alternative pipelines/hyperparameters.

K-Fold Cross-Validation: Procedure



Choosing K: A Practical Trade-Off

The choice of K affects both computation and statistical properties.

Small K (e.g. $K = 3$ or 5):

- faster;
- larger validation folds;
- typically **higher bias** (training sets smaller than full data).

Large K (e.g. $K = 10$):

- more computation;
- training sets closer to full data;
- typically **lower bias** but potentially higher variance of the estimate.

Rule of thumb: $K = 5$ or 10 is often a good compromise in i.i.d. settings.

Model Selection: CV as a Tuning Engine

In practice we compare a **family** of candidates:

$$\{\mathcal{A}_\lambda : \lambda \in \Lambda\},$$

where λ denotes hyperparameters (and possibly feature choices).

We compute $\widehat{R}_{CV}(\mathcal{A}_\lambda)$ for many λ and select:

$$\lambda^\star \;=\; \arg\min_{\lambda \in \Lambda} \widehat{R}_{CV}(\mathcal{A}_\lambda).$$

Important:

- CV is primarily a **model selection** tool;
- the selected model is optimized for the CV criterion, not necessarily for the true unknown risk.

The Hidden Problem: Optimism from Repeated Search

When we try many alternatives, we effectively perform **multiple comparisons**.

Even if every candidate had the same true performance, the best CV score would tend to look artificially good *by chance*.

Consequence:

- CV-based selection can produce an **optimistically biased** estimate of performance if we report the same CV score used for selection.

Takeaway: we must separate **tuning** from **final assessment**.

Nested Cross-Validation (Conceptual Gold Standard)

Nested CV separates:

- **inner loop**: model selection / hyperparameter tuning,
- **outer loop**: unbiased estimate of generalization after selection.

Procedure:

1. Outer split into K_{out} folds.
2. For each outer fold:
   - Use inner CV on the outer-training data to choose $\lambda^{\star}$;
   - Evaluate the chosen model on the held-out outer fold.
3. Average the outer-fold results.

Interpretation: performance estimate accounts for the fact that selection occurred.

Pipelines: CV Must Validate the Whole Workflow

Cross-validation is meaningful only if it evaluates the **entire pipeline**:

Data → Preprocessing → Feature Engineering → Model Training → Prediction

Common leakage pattern:

- fit preprocessing (e.g. scaling, imputation, PCA, feature selection) on the full dataset;
- then run CV on already "informed" features.

Correct principle: every transformation must be **fit only on the training fold** and applied to the validation fold.

Time Series Caveat (Finance): Standard K-Fold Can Be Invalid

Standard K-fold CV assumes observations are approximately **i.i.d.**

Financial data often violate this:

- temporal dependence (autocorrelation),
- non-stationarity and regime changes,
- information flow is time-ordered (causal constraint).

If we randomly shuffle time series data, we may train on "future" information.

Therefore, in finance, CV must respect **time ordering**.

Time-Aware Alternatives (High Level)

For time-dependent data, common CV variants include:

- **Blocked CV**: keep contiguous time blocks together (no random mixing).
- **Expanding window (walk-forward)**: train on past, validate on the next block, then expand.
- **Rolling window**: train on a moving window, validate on the subsequent block.

Key requirement: validation must always occur *after* training in time.

(We will formalize evaluation protocols for finance later, when we discuss backtesting.)

From CV to Model Choice: What to Report

When using CV for model selection, do not report only a single number.

Report at least:

- mean validation performance across folds,
- dispersion across folds (e.g. standard deviation),
- stability of the selected hyperparameters (do we always pick the same region of Λ?).

Interpretation:

- high average + low dispersion suggests stable generalization,
- high dispersion suggests sensitivity to sampling (potential overfitting risk).

What Cross-Validation Fixes . . . and What It Does Not

Cross-validation is designed to reduce dependence on a single split:

- it reduces the **variance of the performance estimate** due to sampling;
- it supports **model selection** under limited data.

But CV does **not** guarantee robustness in finance:

- it does not remove **non-stationarity** (future regimes may differ);
- it does not eliminate **pipeline leakage** unless every step is nested correctly;
- it cannot protect against **selection-induced optimism** when we search extensively.

Takeaway: CV improves **estimation stability**, not **model truth**.

A Classic Failure Mode: “Passes CV, Fails in Production”

A common scenario in finance:

- we tune many models and features until CV performance looks strong;
- the strategy breaks when market conditions shift (volatility regimes, liquidity, policy).

Interpretation:

- the model may have learned **transient structure** specific to the training period;
- the dominating error is not estimator variance, but **distribution shift**.

Operational takeaway:

- time-aware validation and stress tests are essential;
- performance must be judged under **realistic deployment constraints**.

Summary: What CV Is (and Is Not)

Cross-validation is:

- a tool for **model selection**;
- a way to reduce dependence on a single split;
- a framework to validate the **whole pipeline**.

Cross-validation is not:

- a substitute for a final out-of-sample assessment;
- automatically valid for time series without time-aware splits;
- immune to leakage (pipelines matter).

Conclusions

Key Takeaways and Bridge to the Next Lecture

- Training, validation, and test sets serve different roles and must not be confused.
- Model selection (*choose*) and model assessment (*judge*) are distinct goals.
- The test set is a **scientific control**: it should be used once, at the end.
- Leakage often comes from subtle workflow mistakes (preprocessing, feature selection, survivorship).
- In finance, **time ordering is a causal constraint**: evaluation must respect time.
- **Next lecture (Feature Selection & Dimensionality Reduction):**
  - why high-dimensional feature sets amplify overfitting and selection-induced optimism;
  - how to select/reduce features **without leakage** (selection inside the training fold);
  - how dimensionality reduction changes the bias–variance trade-off and affects stability over time.