COMS4995W31
Applied Machine Learning

Dr. Spencer W. Luo

Columbia University | Spring 2026

Announcement


MID-TERM
EXAM

Mid-term 😮

Week 6
Duration: 1.5 hours

Format:
- Multiple Choice ✅
- Short Answer ✏️

Cheat Sheet:
- 1-page A4 (double-sided allowed)
- Handwritten or printed

What You Need to Know 🧠

Course 1 - 5:

- Data prep & feature engineering
- Generative vs. Discriminative models
- Model evaluation & bias - variance
- Ensemble

Focus on:

- Understanding concepts
- Explaining scenarios
- Sharing intuitions

Assignment 1

https://www.gradescope.com/courses/1260107/assignments/7670023

Due: Feb 25, 2026 11:59 PM

Recitation 1

Next week

More details will be released on Ed soon


Coca-Cola

Model Evaluation
Bias-Variance

AI of the Week: The Death of Public Benchmarks 🪦

By early 2026, classic benchmarks like MMLU and GSM8K have become effectively useless

Top models (OpenAI, Claude, Gemini) now score >99% on them, yet often fail in real-world edge cases

The Problem: "Data Contamination"

Since these benchmarks are public on the internet, models have accidentally "trained" on the test questions

The Solution: The rise of Private Leaderboards (e.g., HuggingFace Private). These use dynamic, held-out datasets that model builders never see, ensuring the score reflects true intelligence, not memorization

🔗 Class Connection:

Overfitting

When a model scores 98% on MMLU but fails your specific task, it is overfitting to the benchmark

It has memorized the "test set" (public internet data) instead of learning the underlying pattern

Agenda

- Motivation
- Train / Validation / Test Split
- Common Evaluation Metrics
- Bias - Variance Tradeoff
- Model Selection Strategies

Motivation

Supervised vs. Unsupervised Learning

Supervised Learning

- Learn mapping f: X → Y
- Given data (x_i, y_i)
- Example: Spam Email (label = spam / ham)

Unsupervised Learning

- Discover hidden structures in data
- Only x_i, no labels
- Example: Customer Segmentation

What is Machine Learning (ML)?

- Learn patterns from **data** 🧩
- Make **predictions** on new data 🔮
- **Generalize** beyond training 🌍

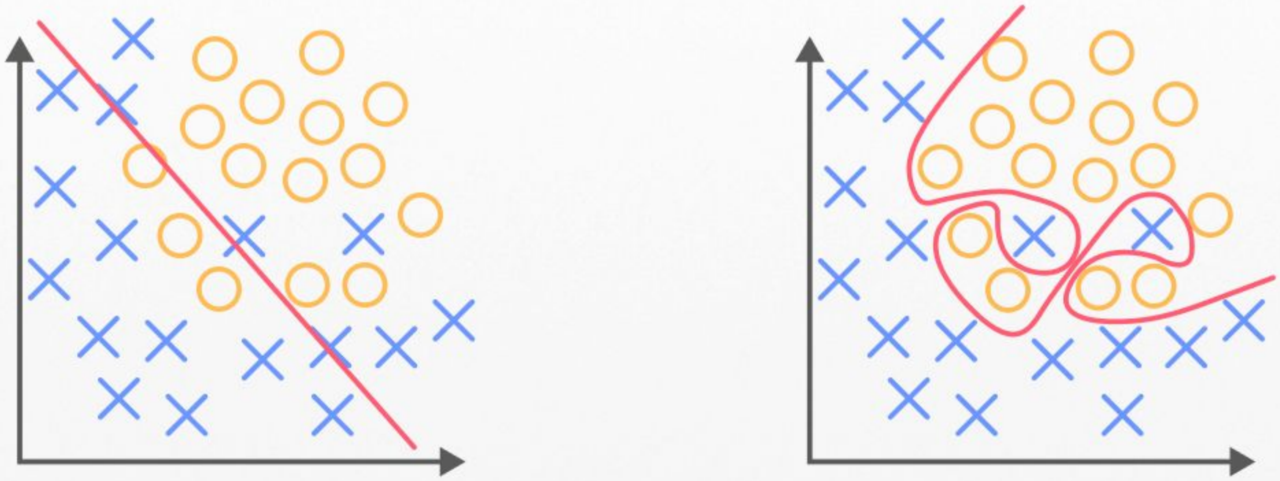
Why Do We Need Model Evaluation? 🤔

Training accuracy ≠ Real-world performance

Without proper evaluation:

- Models may memorize training data (overfitting)
- Models may be too simple to capture patterns (underfitting)

Wrong evaluation → Wrong decisions → Costly mistakes 💸



Underfitting

Overfitting

Overfitting vs Underfitting 📉📈

Overfitting definition:

- Model fits the training data too closely - including random noise
- Symptom:
  - Training Accuracy ≈ 100%
  - Test Accuracy ≪ Training Accuracy

Like a student who:

📖 Memorizes every word from lecture slides (training set)

❌ Cannot answer slightly different questions in mid-term (test set)

Overfitting vs Underfitting 📉📈

Underfitting definition:

- Model is too simple to capture the underlying patterns in data.
- Symptom:
    - Low Training Accuracy
    - Low Test Accuracy

Like a student who:

📖 Barely skimmed the textbook

❌ Cannot even handle problems in the textbook (training set)

❌ Naturally also fails in mid-term (test set)

Why Wrong Metrics Can Be Dangerous 🚨

Accuracy is misleading when data is imbalanced

Example: disease detection dataset

- 99% healthy, 1% sick
- A model that always predicts "healthy" → 99% accuracy but useless ❌

Need metrics that capture what matters: precision, recall, etc.

Key Takeaway 💡

- Evaluation is not optional, it is essential
- Good evaluation = Fair model comparison
- Good evaluation = Reliable deployment in real world

Train / Validation / Test Split

Why Split Data? 🍰

Goal: measure generalization ability

If we only check training accuracy:

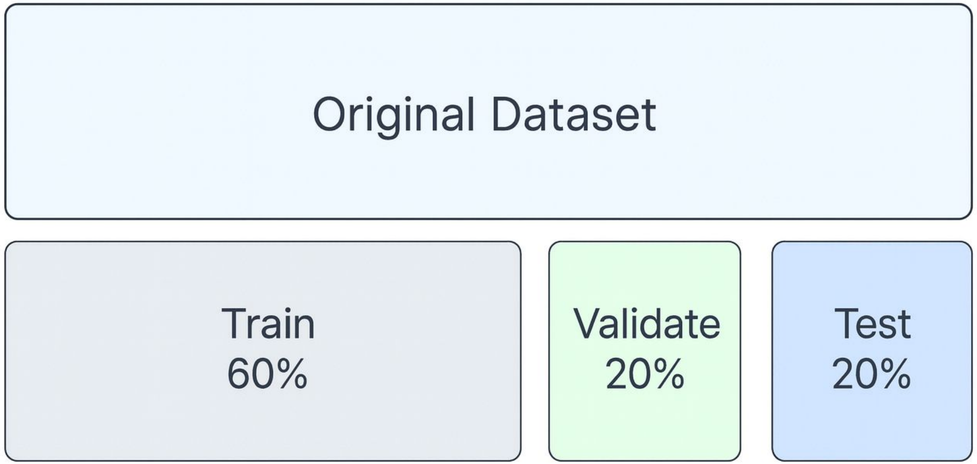
- Might think model is "perfect" → but actually memorizing 📖

Solution: keep separate sets

- Train → learn patterns
- Validation → tune hyperparameters
- Test → simulate unseen real-world data

Why Split Data? 🍰

Multiple Models
Percentage Split



Typical Splits 📊

Common ratios:

- Train: 60 - 80%
- Validation: 10 - 20%
- Test: 10 - 20%

Important rule: Test set is locked 🔒

Hold-Out Method ✂️

One-time split (Train / Val / Test)

- Pros: simple, fast
- Cons:
  - depends heavily on the random split
  - unstable on small data

k-Fold Cross-Validation 🔄

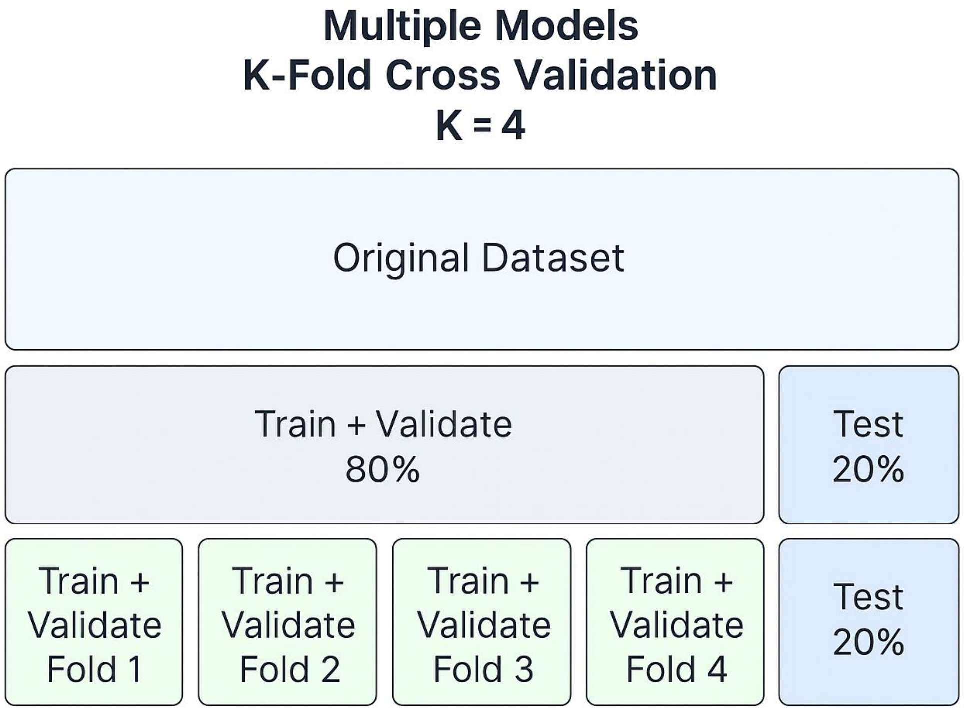
💡 Idea: Rotate the validation set

Steps:

- Split into k equal folds
- Train on (k–1) folds, validate on the remaining one
- Repeat k times, average results

Pros: robust, uses all data
Cons: more compute ⏱️

k-Fold Cross-Validation 🔄



Common Evaluation Metrics

Why Metrics Matter 🎯

Different problems need different metrics

Example:

- Credit card fraud detection → catch rare cases
- Recommender system → focus on ranking quality

⚠️ Accuracy is not enough ❌

Regression Metrics 📉

Mean Squared Error (MSE)

- Penalizes large errors more

$$\mathrm{MSE} = \overbrace{\boxed{\frac{1}{n}\sum_{i=1}^{n}}}^{\text{Mean}}(\overbrace{\boxed{Y_i - \hat{Y}_i}}^{\text{Error}})^{\overbrace{\boxed{2}}^{\text{Squared}}}$$

Regression Metrics 📉

Mean Absolute Error (MAE)

- Less sensitive to outliers

$$MAE = \overbrace{\boxed{\frac{1}{n}}}^{\text{Divide by the total number of data points}} \underbrace{\sum}_{\text{Sum of}} \underbrace{\left| \overbrace{\boxed{y}}^{\text{Actual output value}} - \overbrace{\boxed{\hat{y}}}^{\text{Predicted output value}} \right|}_{\text{The absolute value of the residual}}$$

Classification Metrics ✅/❌

Accuracy

- Proportion of correct predictions
- Simple but misleading on imbalanced data

Confusion Matrix 🔢

		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Recall** $\frac{TP}{(TP + FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN + FP)}$
		Precision $\frac{TP}{(TP + FP)}$	**Negative Predictive Value** $\frac{TN}{(TN + FN)}$	**Accuracy** $\frac{TP + TN}{(TP + TN + FP + FN)}$

Precision, Recall ⚖️

Precision (Purity)

- Out of predicted positives, how many are correct?

Recall (Coverage)

- Out of actual positives, how many did we find?

Example: Disease detection 🧪

- High recall = fewer missed cases
- High precision = fewer false alarms

F1 ⚖️

F1 combines Precision and Recall into one metric

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

👉Different Precision & Recall

A model outputs scores / probabilities, not just hard labels

To decide class = 1, we must set a threshold

- Example: predict positive if score ≥ 0.5

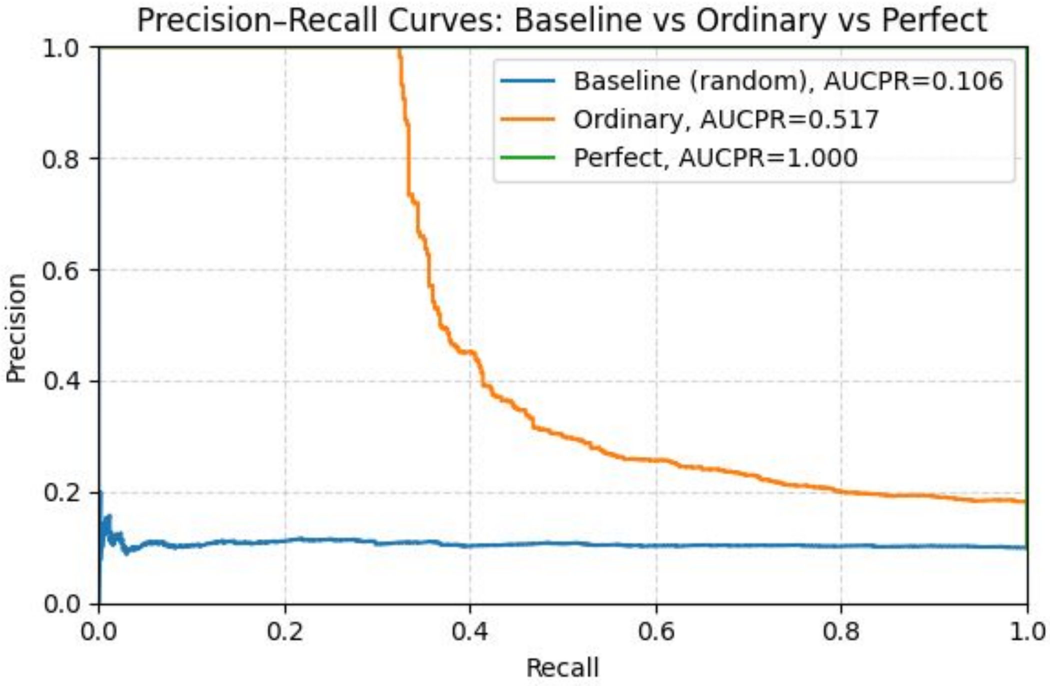
If we change the threshold:

- Lower threshold → more positives predicted → higher Recall, lower Precision
- Higher threshold → fewer positives predicted → higher Precision, lower Recall

⚖️ This trade-off creates multiple (Precision, Recall) pairs

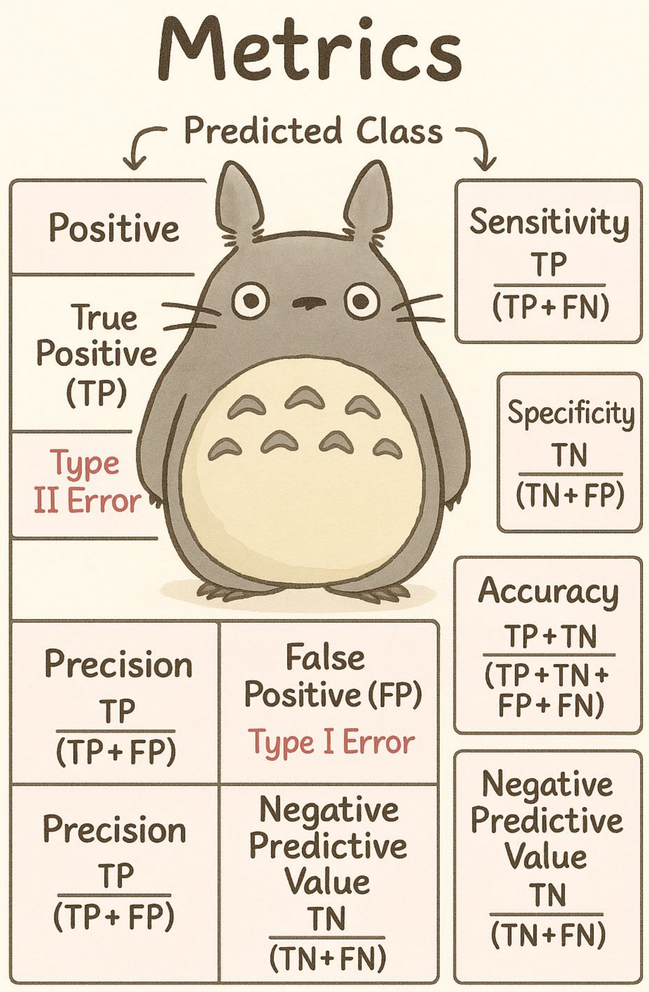
AUCPR 📊

AUC = Area Under Curve



Choosing the Right Metric 🧭

- Regression → MSE, MAE, RMSE
- Balanced classification → Accuracy, F1
- Imbalanced classification → Precision, Recall, F1, AUCPR
- Unsupervised tasks → Normalized Mutual Information (later lectures)


Metrics
Predicted Class
Positive
True Positive (TP)
Type II Error
Sensitivity
TP / (TP + FN)
Specificity
TN / (TN + FP)
Accuracy
TP + TN / (TP + TN + FP + FN)
Precision
TP / (TP + FP)
False Positive (FP)
Type I Error
Precision
TP / (TP + FP)
Negative Predictive Value
TN / (TN + FN)
Negative Predictive Value
TN / (TN + FN)

Bias - Variance Tradeoff

Bias–Variance Decomposition 🎯

Bias

- Error caused by simplifying assumptions in the model, leading to a systematic difference between prediction and ground truth

Symptom

- Model is too simple to capture the true relationship
- Leads to underfitting
- Example: Using a straight line to fit a curved pattern

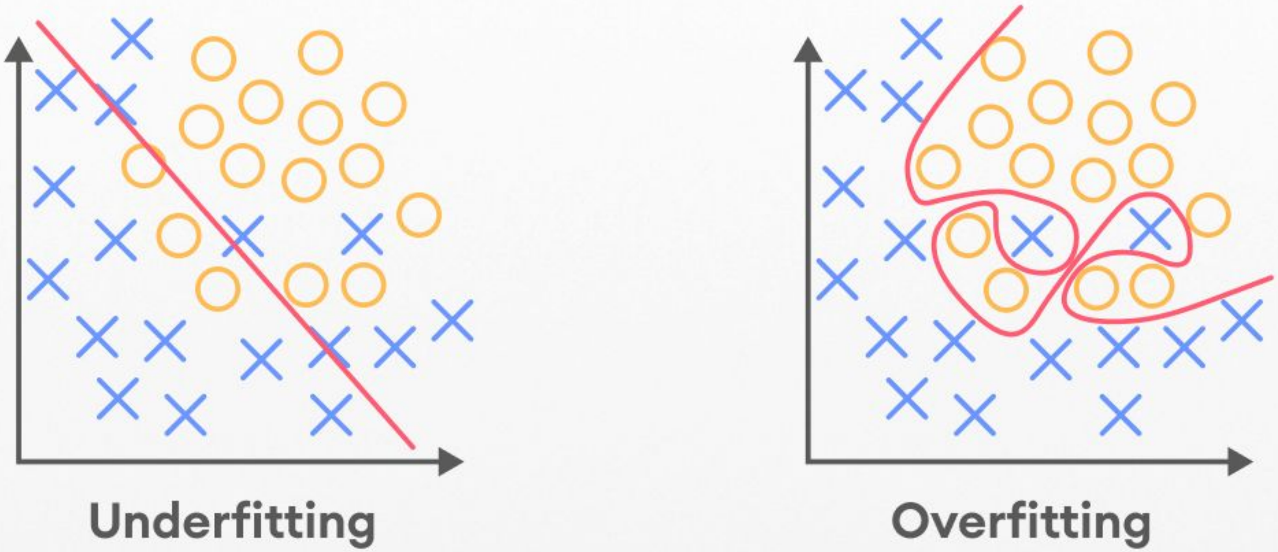
Bias–Variance Decomposition 🎯

Variance

- Error caused by a model's sensitivity to small fluctuations in the training data, leading to inconsistent predictions across different datasets.

Symptom

- Model is too complex, changes a lot with small noises in data
- Leads to overfitting

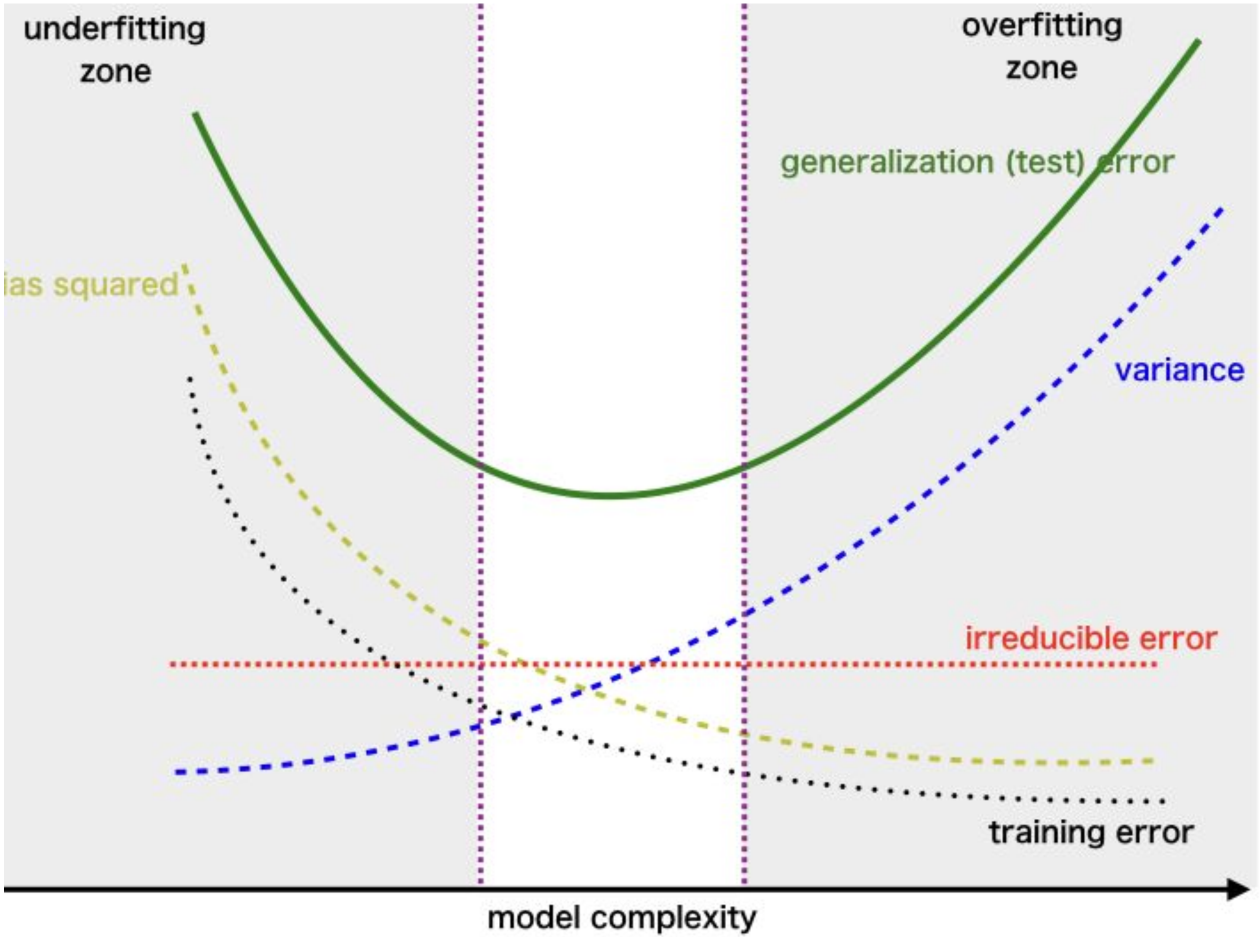

Underfitting
Overfitting

Bias–Variance Tradeoff

Generalization Error = $Bias^2$ + Variance + Noise

- Generalization Error == Expected Test Error

Goal = balance between Bias (too simple → underfit) and Variance (too complex → overfit)



underfitting
zone
overfitting
zone
generalization (test) error
ias squared
variance
irreducible error
training error
model complexity

Bias vs Variance — Key Takeaways 🎯

Opposite failure modes

- High Bias → Underfitting, model too rigid
- High Variance → Overfitting, model too sensitive

Tradeoff is unavoidable

- Reducing bias usually increases variance
- Reducing variance usually increases bias

What we really want

- Minimize expected test error (generalization error)
- Balance both terms + accept some irreducible noise

Model Selection Strategies

Why Model Selection? 🎯

We have mastered:

- How to split data 🍰
- How to measure performance 📊
- Why bias & variance matter ⚖️

Next challenge:

- Choosing the best model among candidates

Cross-Validation for Model Choice 🔄

Not just for evaluation → also for model selection

Use k-fold CV to estimate performance

Pick model with best average validation score

Hyper-parameter Tuning 🔧

Models often have parameters not learned directly

- Regularization strength (α)
- Decision tree depth
- Neural network depth

Strategies

- Grid search 🗂️
- Random search 🎲

Regularization 📉

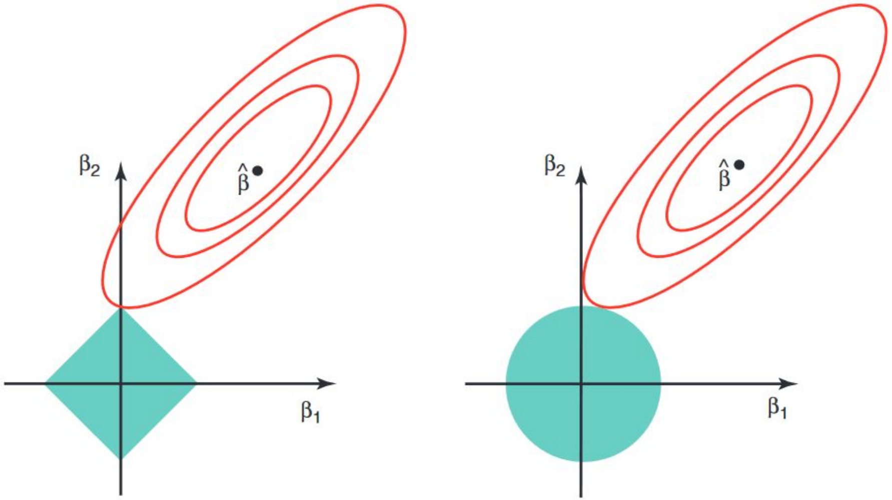
Problem: overly complex models → high variance

Solution: add penalty terms

- Ridge (L2): penalize large weights smoothly
- Lasso (L1): shrink some weights to zero → feature selection

Ridge vs Lasso (Geometry)

Lasso	Ridge
diamond constraint → corners	circular constraint → smooth shrinkage



Early Stopping 🛑

Common in neural networks

Idea:

- Stop training when validation error starts increasing
- Prevents overfitting from too many epochs

Ensemble Methods 🤝

Combine multiple models to reduce variance

Examples:

- Bagging / Random Forests 🌲
- Boosting (AdaBoost, XGBoost) 🚀

Analogy: “wisdom of the crowd”

Practical Workflow 🔄

- Split data into Train / Val / Test
- Define candidate models + hyperparameters
- Use Cross-validation for fair comparison
- Regularize or stop early if variance too high