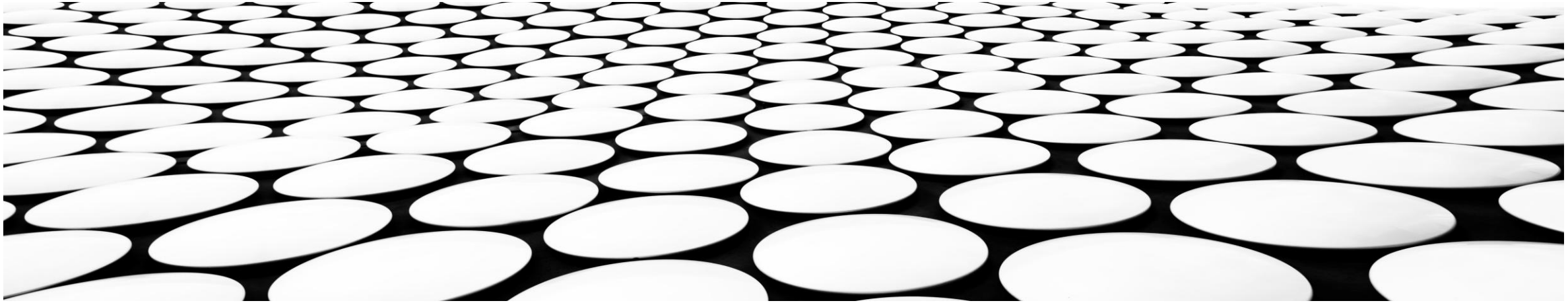
DATA MINING AND PREDICTIVE DATA ANALYTICS

CHAPTER-7

PREPARING TO MODEL THE DATA



SUPERVISED VS. UNSUPERVISED METHODS

- Data mining techniques fall into two broad categories
  - Supervised methods
  - Unsupervised methods
- The classification depends on whether a target variable is specified.

1. Unsupervised Methods

- Unsupervised methods operate without any predefined target variable.
- The algorithm is left to independently discover: natural groupings, hidden patterns, relationships among variables, and underlying structure of the dataset.
- These methods are widely used when we do not know what to predict but want to explore the data.

SUPERVISED VS. UNSUPERVISED METHODS

- **Examples of Unsupervised Methods**
- **Clustering**
  - Groups similar observations into clusters based on similarity measures (distance, density, or statistical distribution).
  - **Applications:** Market segmentation, Customer profiling, Social media community detection, Grouping gene expression profiles, Detection of voter groups for political strategy
  - **Example:** Political analysts may apply clustering on socioeconomic and demographic variables like: age, gender, education, income, geographic location to discover coherent voter clusters without any predefined label which may help in targeted campaign strategies.

SUPERVISED VS. UNSUPERVISED METHODS

- Association Rule Mining:
  - Uncover interesting relations among variables in large datasets.
- Example:
  - **Market Basket Analysis**: Retailers want to know which items tend to be purchased together. No target variable exists; instead, the focus is on co-occurrence relationships.
  - **Challenges:** Millions of possible item combinations → combinatorial explosion. Requires specialized algorithms like Apriori, FP-Growth, etc.
  - **Outcome:** Helps in store layout optimization, Product bundling, Cross-selling and recommendations

SUPERVISED VS. UNSUPERVISED METHODS

- Key Characteristics of Unsupervised Methods
  - No training labels or output variable
  - Explore hidden structures without predefined targets.
  - Focus on discovering structure rather than prediction
  - Results often require interpretation and domain expertise
  - Useful for preprocessing before supervised learning (e.g., aggregation, dimensionality reduction)

SUPERVISED VS. UNSUPERVISED METHODS

2. Supervised Methods

- Supervised techniques involve learning from examples where both: Input variables (predictors) and Output variable (target) are known.
- The goal is to learn a relationship or function that maps inputs to outputs.

- **Examples of Supervised Method:**
- **Regression:**
  - These are used when the target variable is continuous.
  - Example: Tasks Predicting house prices, Estimating medical costs, Forecasting sales or revenue.
  - Mechanism: The least squares algorithm adjusts the model to minimize the difference between the actual output (observed y) predicted output (ŷ)

SUPERVISED VS. UNSUPERVISED METHODS

- Classification
  - Used when the target variable is categorical.
  - Examples: Spam vs. non-spam e-mails, Disease diagnosis, Fraud detection, Image classification
  - Techniques: Decision Trees, Random Forests, Neural Networks, k-NN, SVM
  - The model is trained on labelled data, enabling it to make predictions on unseen samples.

SUPERVISED VS. UNSUPERVISED METHODS

- Key Characteristics of Supervised Methods
  - Known target variable
  - Rely on labeled data to make predictions.
  - Model learns from examples
  - Focus is on prediction accuracy
  - Requires large labeled datasets
  - Performance can be measured using metrics (accuracy, RMSE, precision, recall)

SUPERVISED VS. UNSUPERVISED METHODS

Feature	Supervised	Unsupervised
Target variable	Present	Absent
Primary goal	Predict known outcomes	Discover hidden structure
Data requirement	Labeled data	Unlabeled data
Typical methods	Regression, classification	Clustering, association rules
Performance evaluation	Prediction error, accuracy	Cluster cohesion, rule interestingness
Nature of analysis	Often model-driven	Often exploratory

STATISTICAL METHODOLOGY AND DATA MINING METHODOLOGY

- Statistical methodology and data mining methodology differ in the following two ways:

1. Applying statistical inference using the huge sample sizes encountered in data mining tends to result in statistical significance, even when the results are not of practical significance.

2. In statistical methodology, the data analyst has an a priori hypothesis in mind. Data mining procedures usually do not have an a priori hypothesis, instead freely trolling through the data for actionable results.

STATISTICAL METHODOLOGY AND DATA MINING METHODOLOGY

- Statistical methodology and data mining methodology differ in the following two ways:
- 1. Role of Sample Size:
  - With very large sample sizes (common in data mining) Even extremely small differences or correlations can become statistically significant.
  - But significance does not imply practical importance.
  - This creates a need to emphasize practical relevance, and business value in data mining applications.
- 2. A Priori Hypothesis vs. Exploratory Search
- A. Statistical Methodology
  - Typically begins with an a priori hypothesis (based on theory or prior knowledge).
  - Objective: confirm or reject the hypothesis using statistical tests.
  - Examples:Determining if two population means differ. Estimating a regression coefficient with confidence intervals

STATISTICAL METHODOLOGY AND DATA MINING METHODOLOGY

- B. Data Mining Methodology:
  - Often does not start with a defined hypothesis.
  - The analyst explores data in an open-ended manner:
    - discovering patterns
    - mining associations
    - building predictive models
  - Data mining is action-oriented: find results that can improve decisions.
  - Risk: Exploratory search may lead to:
    - Overfitting
    - spurious patterns
    - false discoveries arising purely by chance
  - Thus proper validation (e.g., cross-validation, hold-out sets) is essential.

CROSS-VALIDATION

- Why Cross-Validation Is Needed
  - In data mining, one major risk is data dredging (also known as overfitting or finding spurious patterns).
  - This happens when an analyst mistakenly discovers patterns that:
    - exist only due to random chance,
    - do not generalize to new data,
    - or are not truly meaningful.
  - Cross-validation is therefore essential to ensure that patterns discovered in the training data will also hold for unseen, future data.

CROSS-VALIDATION

- What Is Cross-Validation?
  - Cross-validation is a technique used to evaluate how well an analytical model generalizes to new data.
  - It involves:
    - splitting the available dataset into two or more parts,
    - building the model on one part (training set),
    - testing its performance on the other part(s) (test set).
  - This allows us to estimate the model's ability to perform on unseen data.
  - In data mining, the most common methods are **twofold cross-validation** and **k-fold cross-validation.**

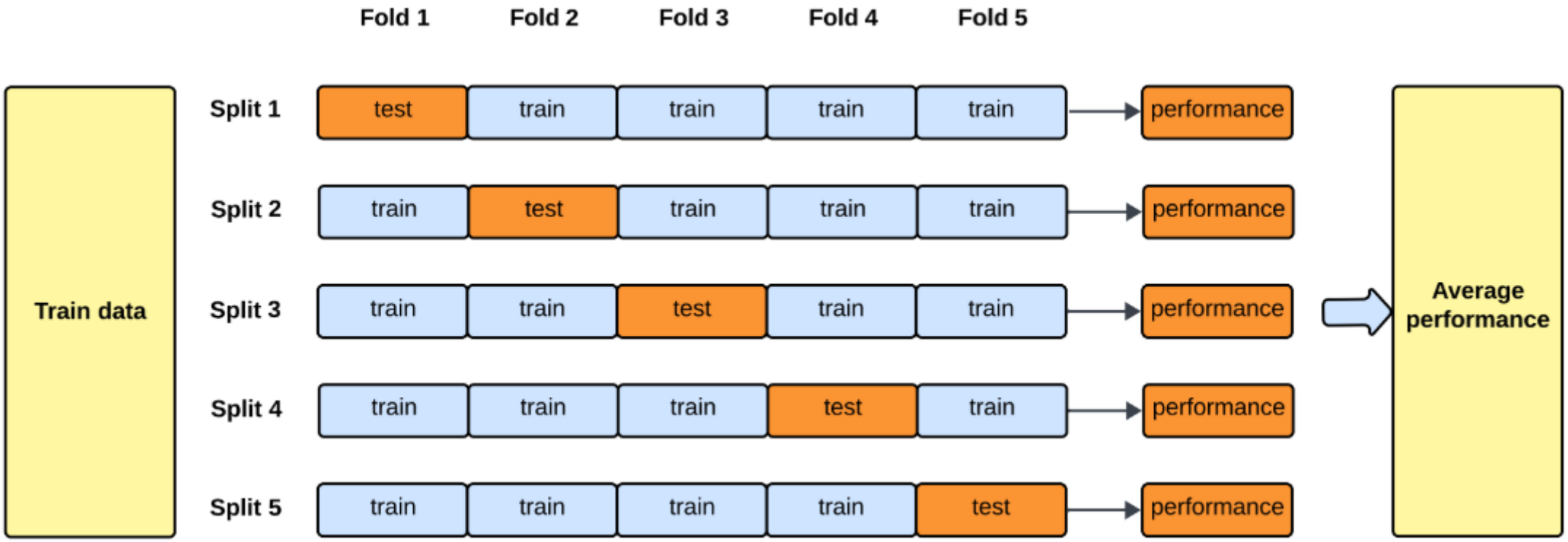
CROSS-VALIDATION

- **Twofold Cross-Validation**
  - Data is randomly split into two sets: **Training Set** and **Test Set**
  - The model is trained on the training set and evaluated on the test set.
  - Key Requirement: The test set must not contain the target variable during model building.
  - The test set should only be used after the model is built.
  - The model built from the training set is then used to predict the hidden target values in the test set.
  - The efficacy of the classifications is then evaluated by comparing predicted values against the true values of the target variable.
  - The provisional data mining model is then adjusted to minimize the error rate (or increase the prediction/classification accuracy) on the test set.

CROSS-VALIDATION

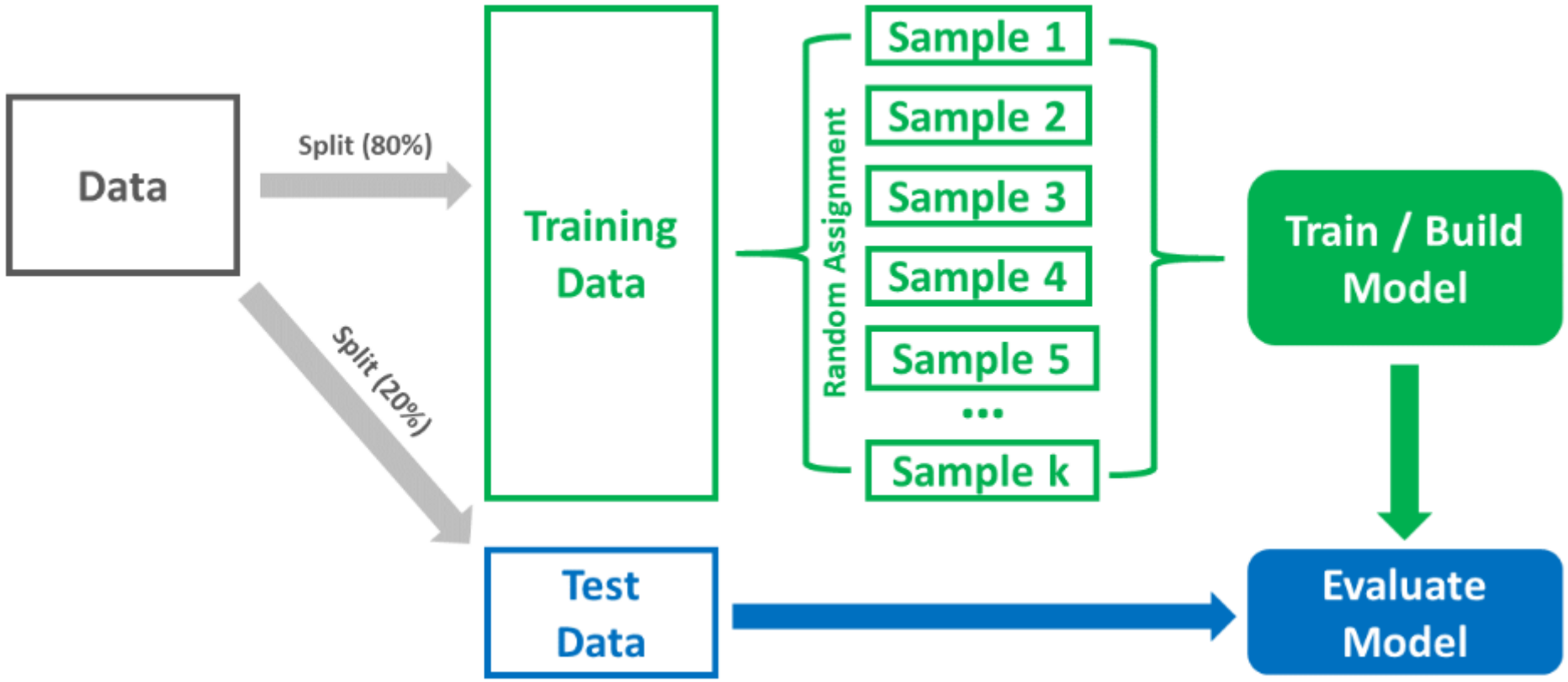
- k-Fold Cross-Validation
  - The dataset is divided into k subsets (folds).
  - The process:
    - Use k−1 folds as training set.
    - Use the remaining one fold as the test set.
    - Repeat the process k times, each time using a different fold as the test set.
    - The results from the k models are then combined using averaging or voting.
  - Common Choice k = 10 (10-fold CV) is widely used
  - Advantages:
    - Every record becomes a test observation exactly once.
    - Evaluation is more stable because it averages over multiple splits.
  - Drawback: Requires more computation since the model must be trained k times.

CROSS-VALIDATION



5-Fold Cross Validation

CROSS-VALIDATION



CROSS-VALIDATION

- Validating the Training–Test Partition
  - To ensure reliable model evaluation, the data analyst must confirm that the training and test datasets are truly independent and comparable (This process is known as validating the partition).
  - Even with random splitting, imbalances may occur.
  - For example, a larger proportion of positive cases of an important flag variable might end up in the training set compared to the test set.
  - Such imbalance can:
    - bias the model during training,
    - create misleadingly high accuracy,
    - reduce predictive performance on new data.
  - Therefore, it is crucial that the distribution of the target variable is as similar as possible in both datasets.

CROSS-VALIDATION

- Methodology For Building And Evaluating A Data Model

1. Partition the available data into a training set and a test set. Validate the partition.
2. Build a data mining model using the training set data.
3. Evaluate the data mining model using the test set data.

Note: How to Validate the Partition?

- Use statistics based hypothesis test (Chapter 6) for validating the target variable, based on the type of target variable.

OVERFITTING

- **Goal of supervised learning:** to develop a model that generalizes well to unseen test or validation data, not just the training set.
- A common challenge is **overfitting**, where the model becomes too closely tailored to the training data.
- In overfitting,
  - The model learns noise and random fluctuations and unique or rare patterns in the training data instead of true underlying patterns.
  - As a result, it achieves very high training accuracy but performs poorly on new data.
  - This usually occurs when the model is overly complex and memorizes training examples rather than learning generalizable relationships.

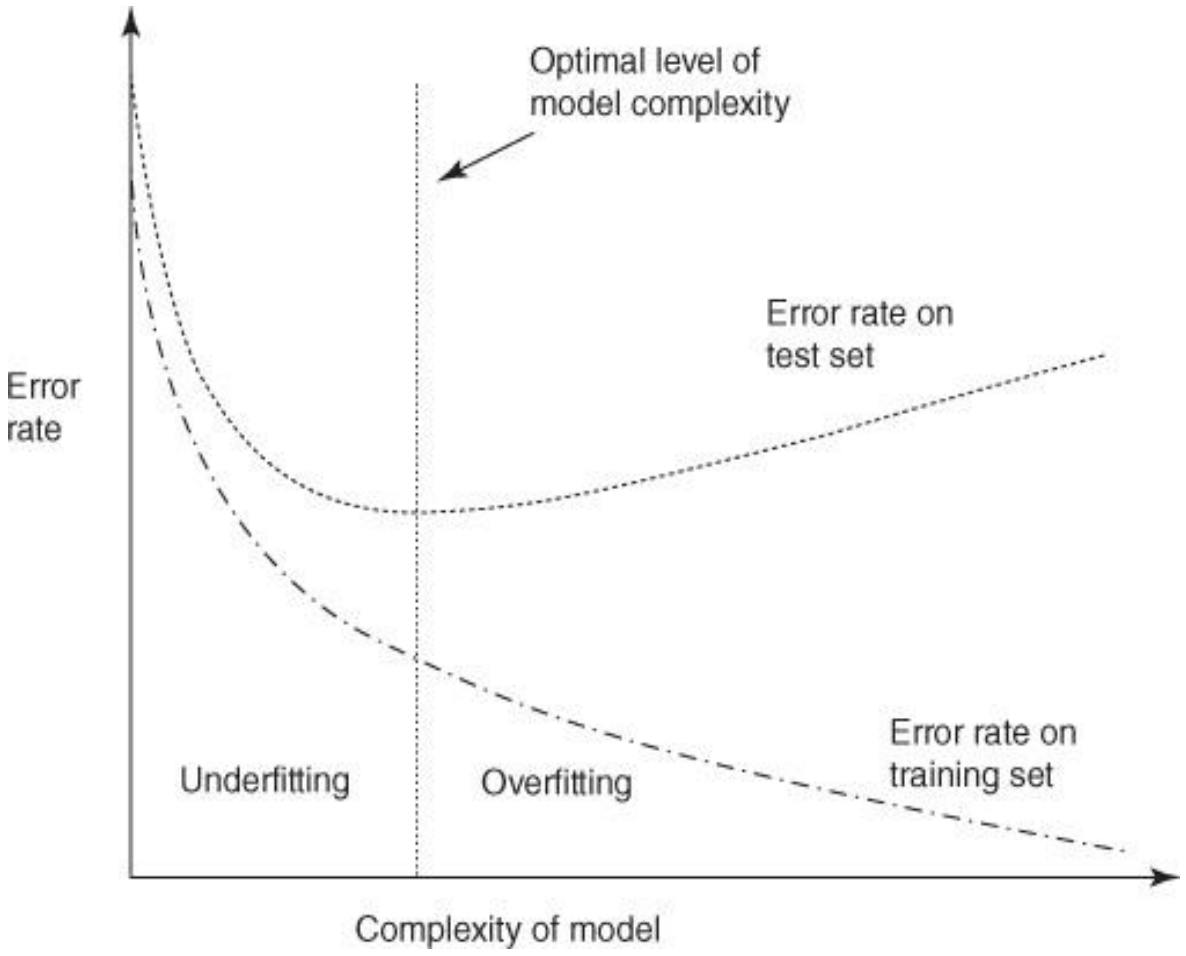
OVERFITTING

Why Does Overfitting Occur?

- Overfitting usually happens due to excessive model complexity, such as:
  - Using too many parameters
  - Using overly flexible model (deep trees, high-degree polynomial functions, very deep NN)
  - Training for too many epochs
- As the model grows in complexity, it tries to account for every possible variation—even rare cases in the dataset (thereby reducing generalizability)
- While this reduces training error, it increases test error.
- This creates the **classic trade-off** in machine learning: **Model complexity vs. generalizability**
- A more complex model fits the training data very well but risks poor performance on unseen data.

OVERFITTING

- Interpreting the Error vs. Complexity Curve



❑ Underfitting (Low Complexity)

When the model is too simple:

- It cannot capture important patterns.
- Both training and test errors are high.
- This is called **underfitting**.

❑ Increasing Complexity

- As complexity increases:
- Training error decreases.
- Test error also decreases initially, as the model starts learning meaningful structure.

❑ Overfitting (High Complexity)

Beyond a certain point:

- Training error keeps falling.
- Test error begins to rise.
- This happens because the model starts memorizing noise or rare details in the training data, losing its ability to generalize.

OVERFITTING

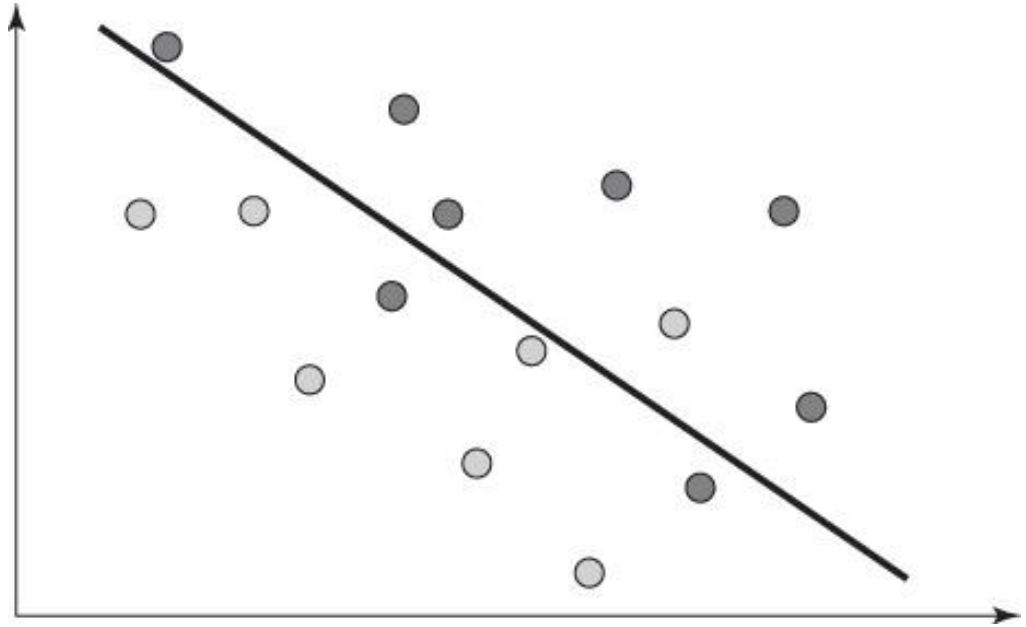
- **Optimal Model Complexity**
  - Between underfitting and overfitting lies an important point
  - Optimal Complexity: The point where the test error is minimal.
  - This is the ideal balance between bias and variance.
  - At this point:
    - The model captures the underlying structure of the data.
    - It generalizes well to unseen test data.
    - It has not yet begun to memorize noise.
  - Going beyond this point causes overfitting; staying below it causes underfitting.
  - Models should aim for this optimal complexity to generalize well.

BIAS–VARIANCE TRADE-OFF

- In supervised learning, selecting an appropriate model complexity is crucial for achieving good predictive performance.
- The bias–variance trade-off explains the fundamental tension between:
  - fitting the training data well, and
  - generalizing effectively to unseen data.
- This concept helps us understand:
  - underfitting, overfitting, and
  - why neither overly simple nor overly complex models are ideal.

BIAS–VARIANCE TRADE-OFF

- Low-Complexity Models and High Bias

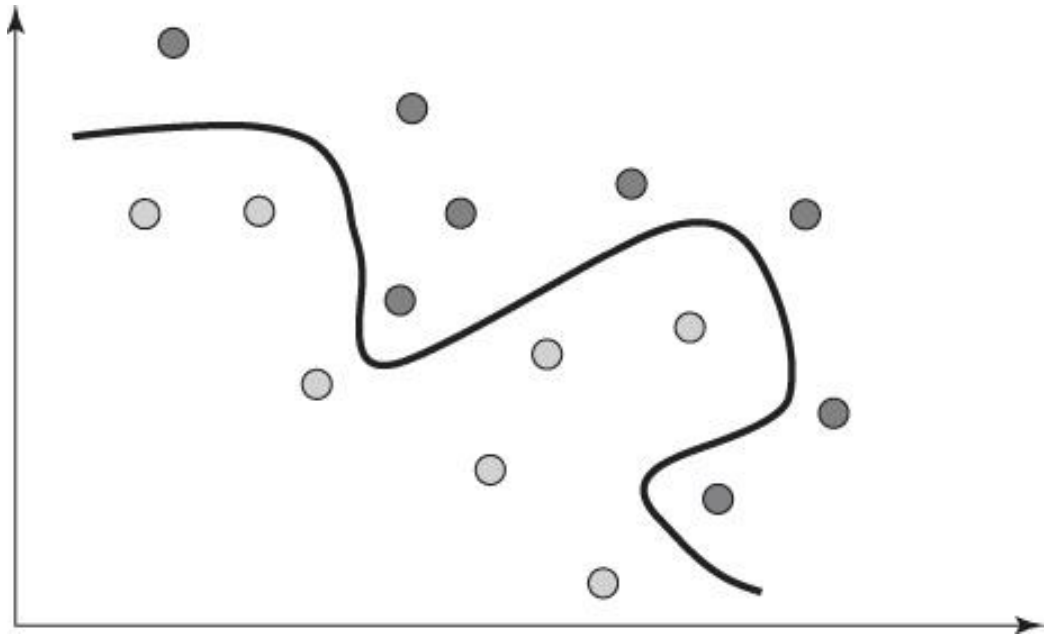


scatter plot to separate two classes (dark gray and light gray points) using a **straight line**.

- A **low-complexity model**, such as a straight-line separator, is simple and stable but misclassifies several points.
- Because it cannot capture important data patterns, it **has high bias** and relatively high training error.

BIAS–VARIANCE TRADE-OFF

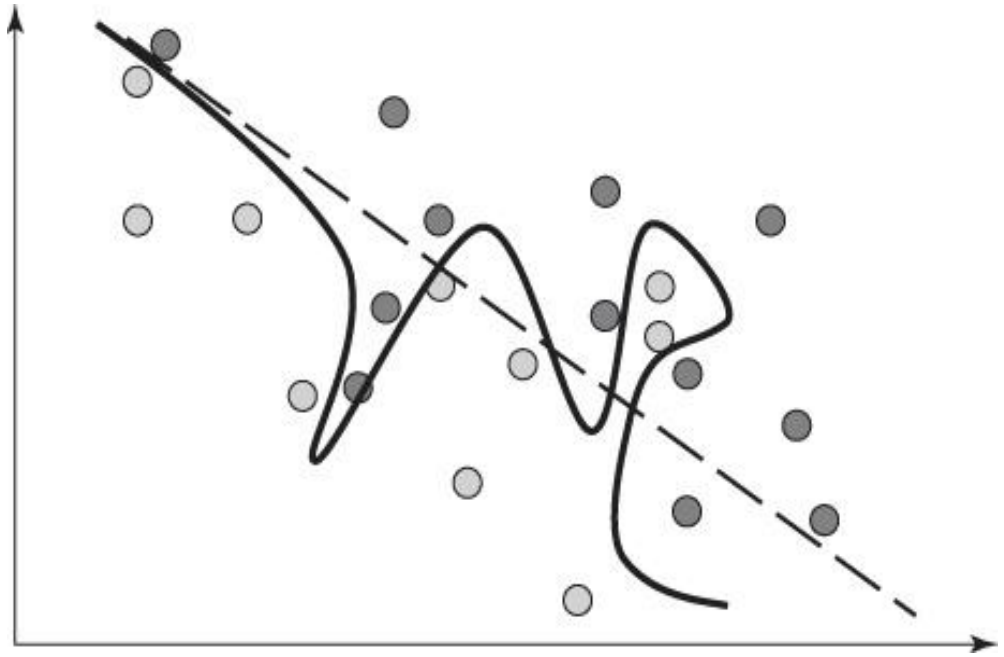
- High-Complexity Models and Low Bias



- a **high-complexity model**, such as a flexible curved separator can fit the training data extremely well and even achieve zero training error.
- This indicates **low bias**, as the model closely adapts to the training data.
- However, such models **risk overfitting** by learning noise and data-specific irregularities rather than general patterns.

BIAS–VARIANCE TRADE-OFF

- Effect of New Data: Variance
  - Variance measures how much a model's predictions change when trained on different samples of data



scatter plot to separate two classes after adding new data

❑ **Low-Complexity Model**

- The straight-line separator changes very little.
- This indicates **low variance**.
- The model is stable with respect to changes in data.

❑ **High-Complexity Model**

- The curved separator must change substantially to accommodate new points.
- This indicates **high variance.**
- The model is sensitive to small changes in the dataset.

BIAS–VARIANCE TRADE-OFF

- Bias–Variance Trade-Off and Overfitting
  - **Low-complexity models –**
    - have high bias and low variance, often leading to underfitting
  - **High-complexity models-**
    - have low bias and high variance, leading to overfitting.
  - **As model complexity increases**-
    - bias decreases but variance increases.
  - **The objective -** select a model that balances both, achieving good generalization without excessive bias or variance.

BIAS–VARIANCE TRADE-OFF

- **Mean Squared Error (MSE) and the Trade-Off**
  - For continuous target variables (Regression or Prediction), Mean Squared Error (MSE) is a commonly used evaluation metric.
  - Why MSE is Important
    - MSE captures both components of error Bias Variance
    - It can be expressed as:

$$\text{MSE} = \text{Variance} + \text{Bias}^2$$

    - This equation highlights that minimizing prediction error requires balancing both bias and variance, not focusing on only one.

BALANCING THE TRAINING DATA SET

- **Motivation for Balancing the Training Data**
  - In many classification problems, one target class occurs far less frequently than others.
  - Such datasets are called imbalanced datasets.
  - If left unbalanced, classification algorithms tend to favor the majority class, leading to misleadingly high accuracy but poor real-world usefulness.
- Example: Fraud Detection (Two-Class Classification)
  - Total sample transactions: 100,000
  - Fraudulent transactions: 1,000 (1%)
  - A classifier predicting “non-fraudulent” cases achieves 99% accuracy, yet fails to detect fraud.
  - This demonstrates why balancing the training data set is often necessary.

BALANCING THE TRAINING DATA SET

- Methods for Balancing the Training Data

1. Resampling the Rare Class (Oversampling)

- Randomly sample minority-class records with replacement.
- Increases the relative frequency of rare outcomes (minority-class) without discarding majority data.

2. Reducing the Majority Class (Undersampling)

- Randomly discard a number of majority-class (non-rare) records.

BALANCING THE TRAINING DATA SET

- Example on Resampling (Oversampling)
  - Suppose we are building a fraud detection classifier.

Class	Number of Records
Non-Fraud (Majority)	9,900
Fraud (Minority)	100
Total	**10,000**

- Fraud cases = 1%
- If the model predicts “Non-Fraud” always → 99% accuracy
- Useless for fraud detection

  - Goal of Resampling
    - We want the model to see enough fraud cases to learn fraud patterns.
    - Let us target a 20% fraud proportion in the training data.

BALANCING THE TRAINING DATA SET

- **Step 1- Compute How Many Fraud Records to Add:**
  - General Formula:

$$x = \frac{p(\text{records}) - \text{rare}}{1 - p}$$

  - Substitute values:

$$x = \frac{0.20(10,000) - 100}{1 - 0.20}$$

$$x = \frac{2,000 - 100}{0.80} = \frac{1,900}{0.80} = 2,375$$

where:

- x = additional resampled records
- p = desired proportion of rare class
- *records* = total records in original data
- *rare* = original number of rare-class records

BALANCING THE TRAINING DATA SET

- Step 2-Perform Resampling
  - Randomly select fraud records with replacement
  - Add 2,375 duplicate fraud records
  - New Training Data:

Class	Records After Resampling
Non-Fraud	9,900
Fraud	100 + 2,375 = 2,475
Total	**12,375**

  - Check balance:

$$\frac{2,475}{12,375} = 0.20\ (20\%)$$

  - Desired balance achieved

BALANCING THE TRAINING DATA SET

- What “With Replacement” Means (Important)
  - The same fraud case may appear multiple times
  - No new fraud cases are to be invented
  - We are reinforcing existing fraud patterns, not fabricating data
- Why This Helps the Model
  - The model now sees fraud patterns repeatedly
  - Fraud no longer gets ignored during training
  - The classifier learns meaningful decision boundaries

Original Fraud IDs	Resampled Fraud IDs
F12	F12
F27	F27
F12	F12 (again)
F88	F88
F27	F27 (again)

BALANCING THE TRAINING DATA SET

- Undersampling
  - Balances data by randomly removing some of the majority-class records.
  - Retains only a subset of majority data to increase minority-class proportion.
  - Advantage: Reduces training data size and computational cost, making model training faster and simpler.
  - Limitations: Valuable information may be lost by discarding majority-class records, which can reduce model accuracy and robustness if the retained data is not sufficiently representative.