الجامعة السورية الخاصة
SYRIAN PRIVATE UNIVERSITY

Topic 4

كلية الهندسة

الذكاء الصنعي العملي

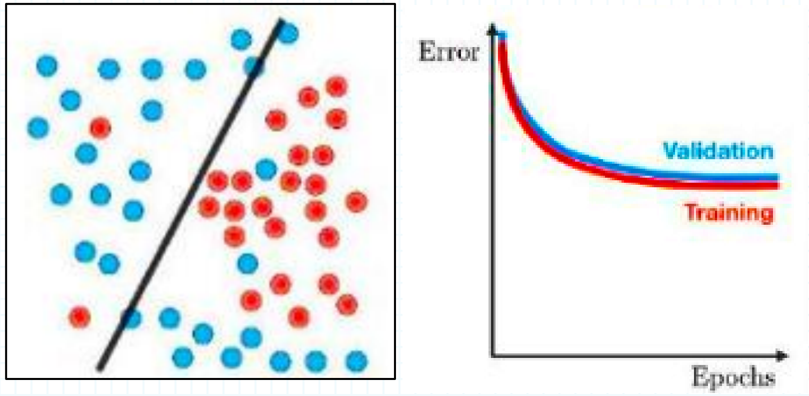
Practical Concerns for Machine Learning

د. رياض سنبل

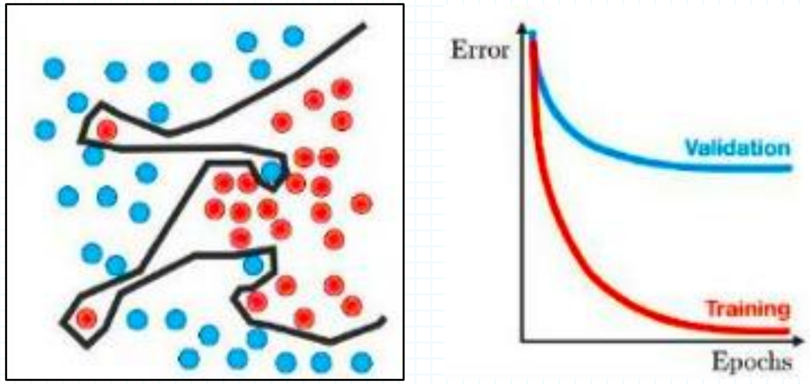
Generalization (Bias-Variance Tradeoff)

Generalization

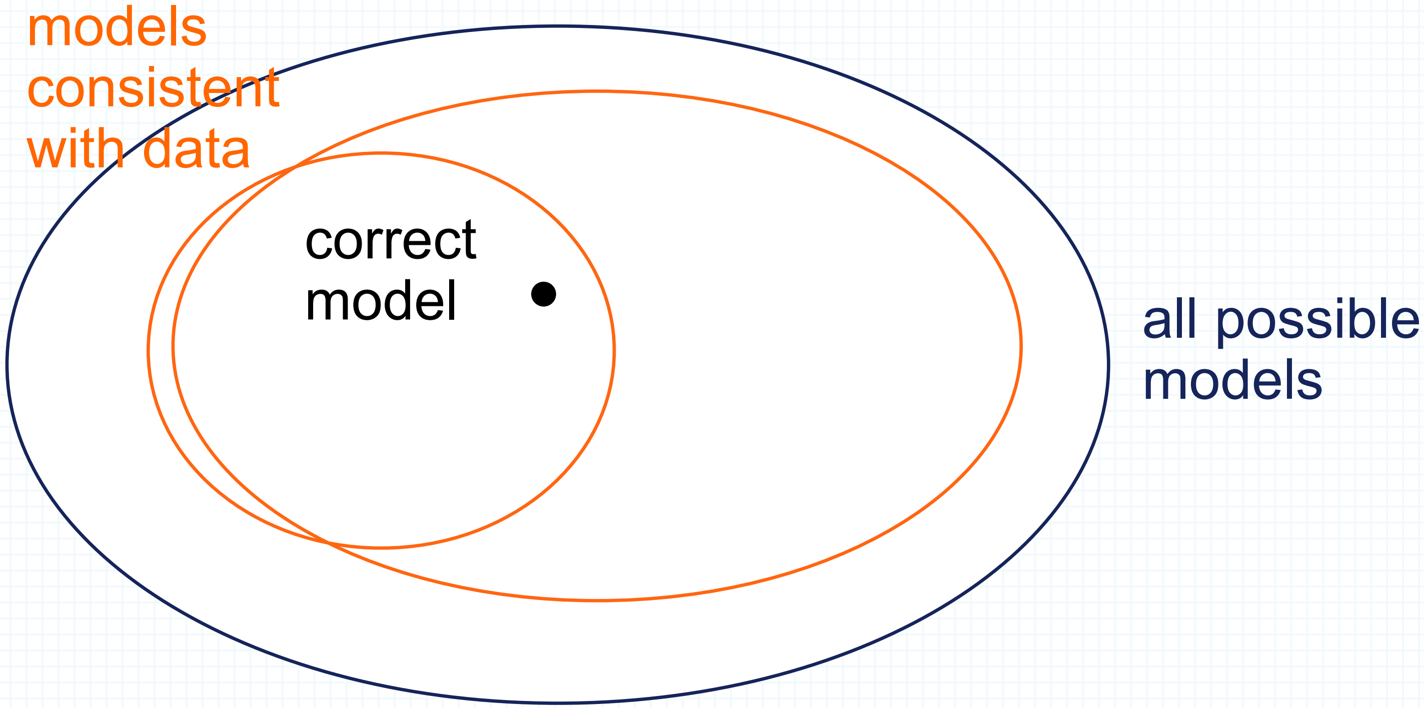
- ***Underfitting***
  - The model is too “simple” to represent all the relevant class characteristics
  - E.g., model with too few parameters produces high error on the training set and high error on the validation set



- ***Overfitting***
  - The model is too “complex” and fits irrelevant characteristics (noise) in the data
  - E.g., model with too many parameters produces low error on the training set and high error on the validation set

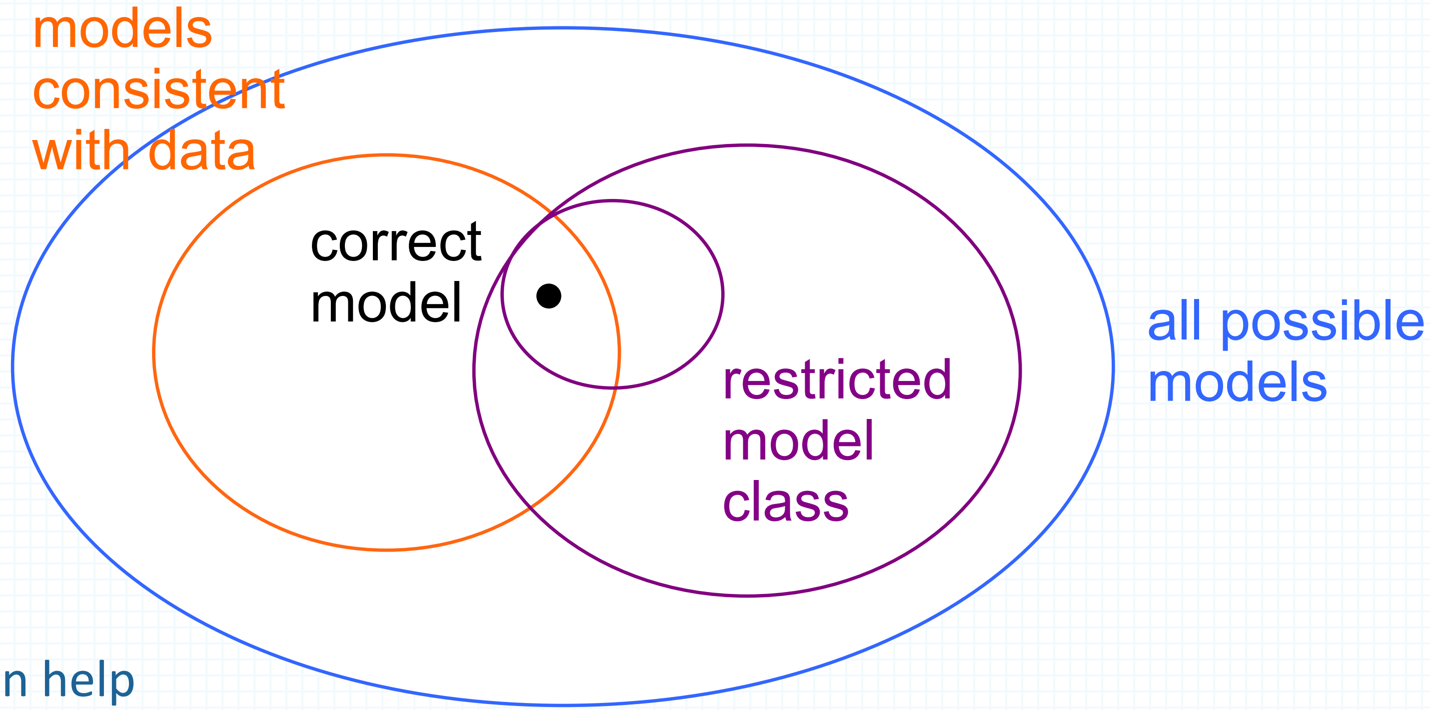


Back to the core idea in ML☺



More data can help!

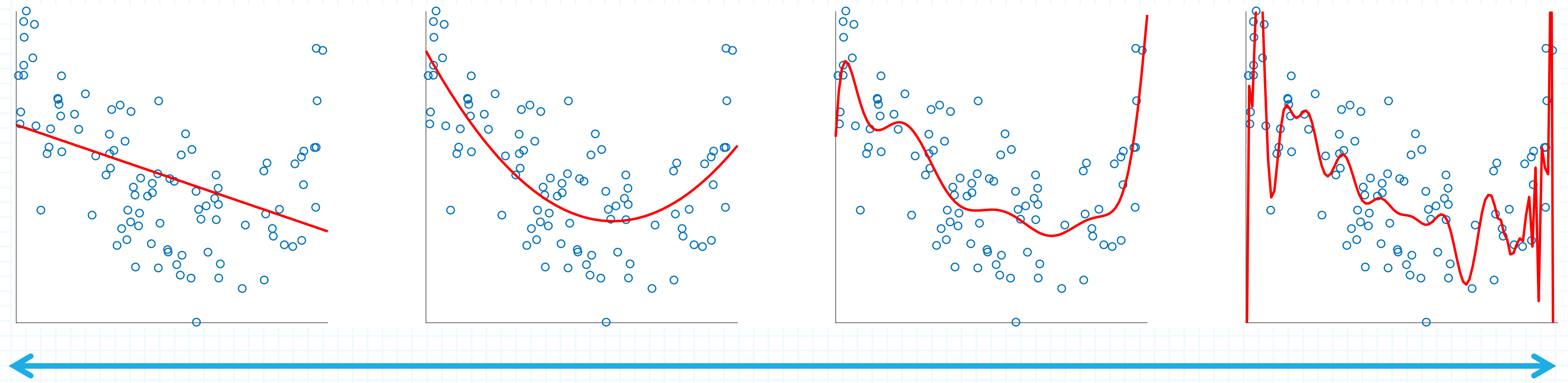
Back to the core idea in ML☺



- Restricting model class can help
- Or it can hurt
- Depends on whether restrictions are domain appropriate

Restricting Models

- Models range in their flexibility to fit arbitrary data



simple model

constrained

small capacity may prevent it from representing all structure in data

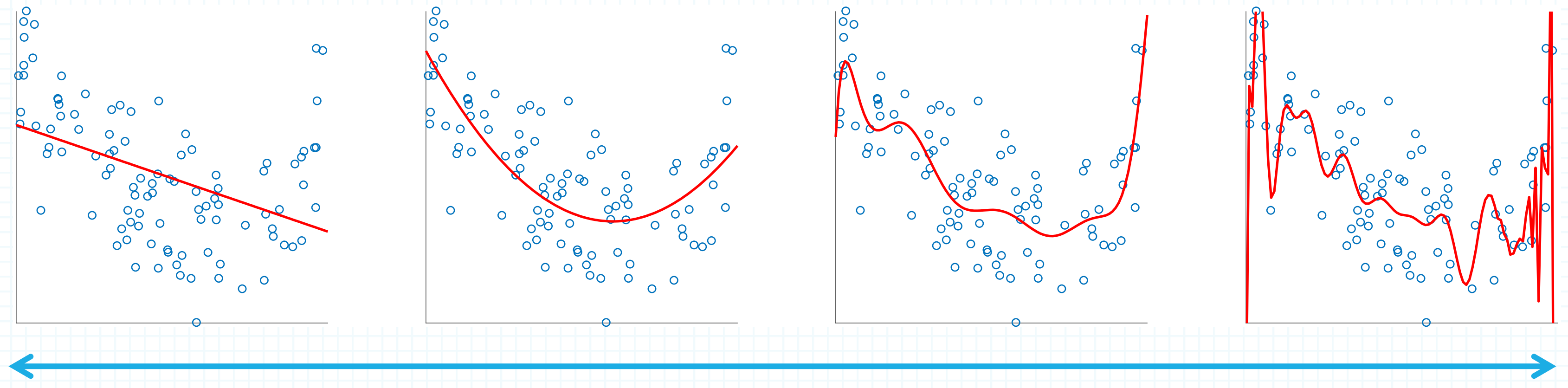
complex model

unconstrained

large capacity may allow it to fit quirks in data and fail to capture regularities

Bias vs Variance

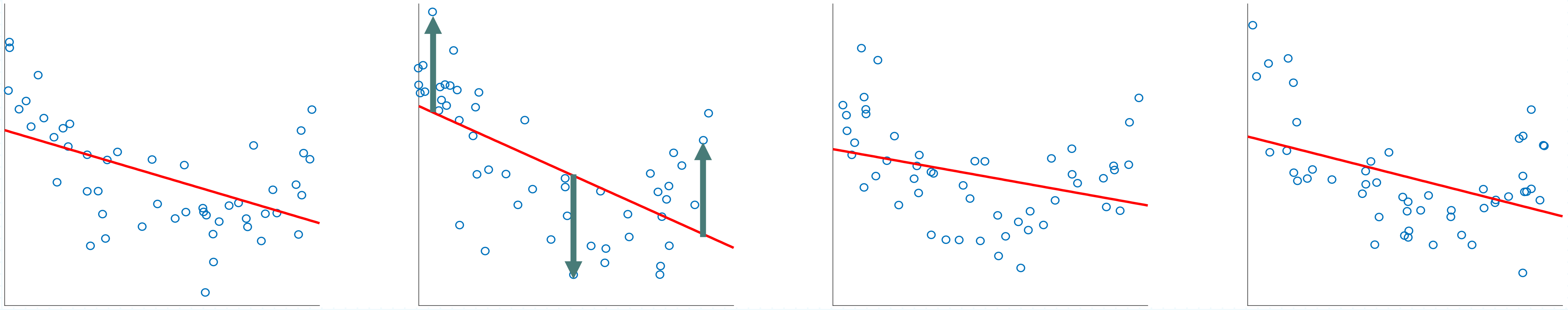
- Models range in their flexibility to fit arbitrary data



high bias *low bias*

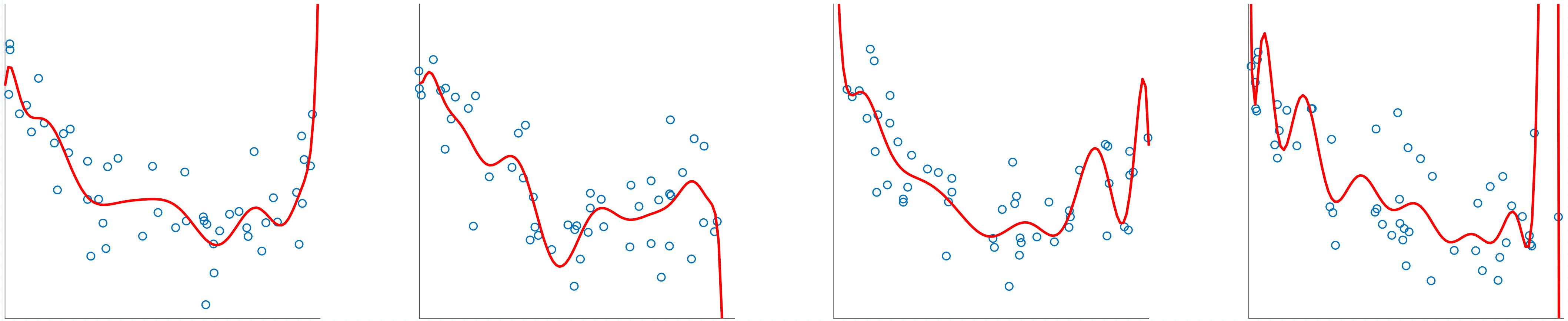
low variance *high variance*

Bias



- Bias is the amount that a model's prediction differs from the target value, compared to the **training data**.
- Bias error **results from** <u>simplifying</u> the assumptions used in a model so the target functions are easier to approximate.
- Regardless of training sample, or size of training sample, model will produce consistent errors

Variance

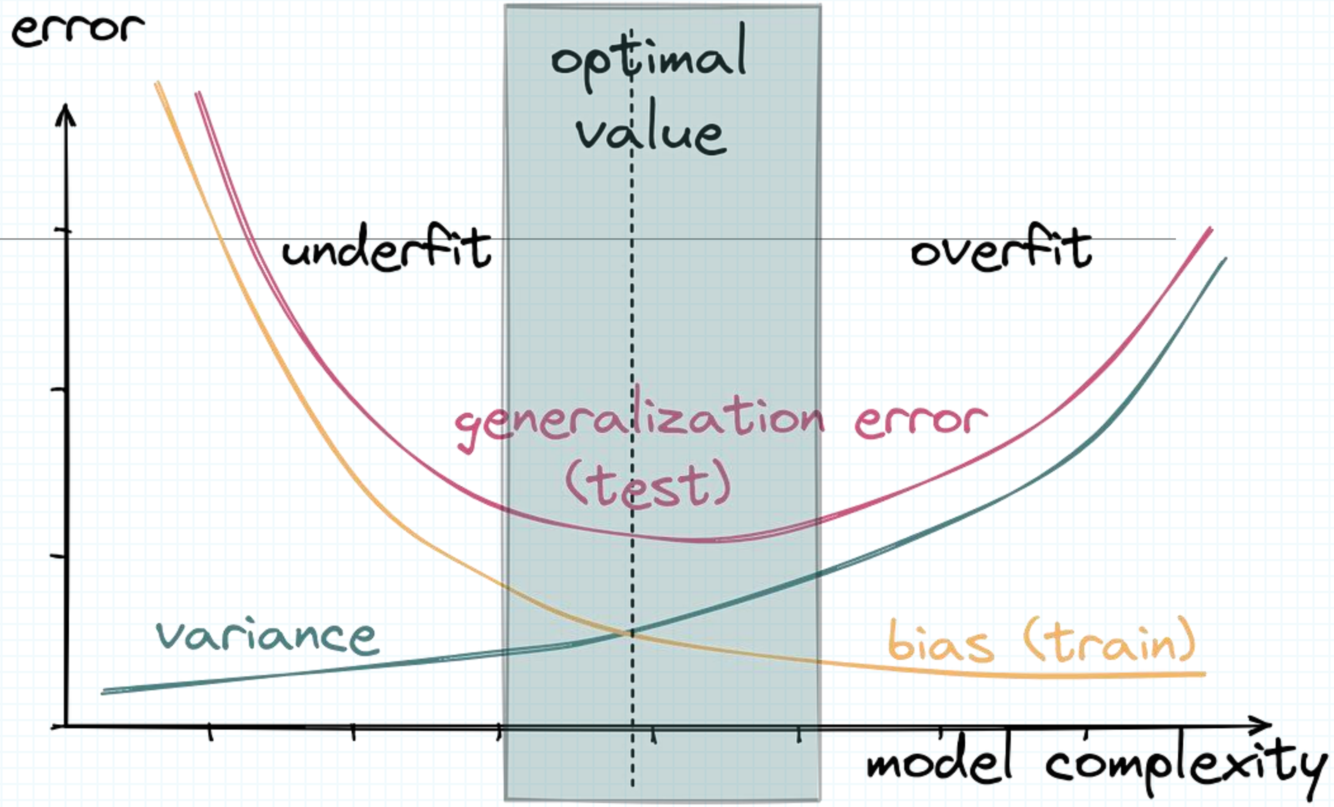


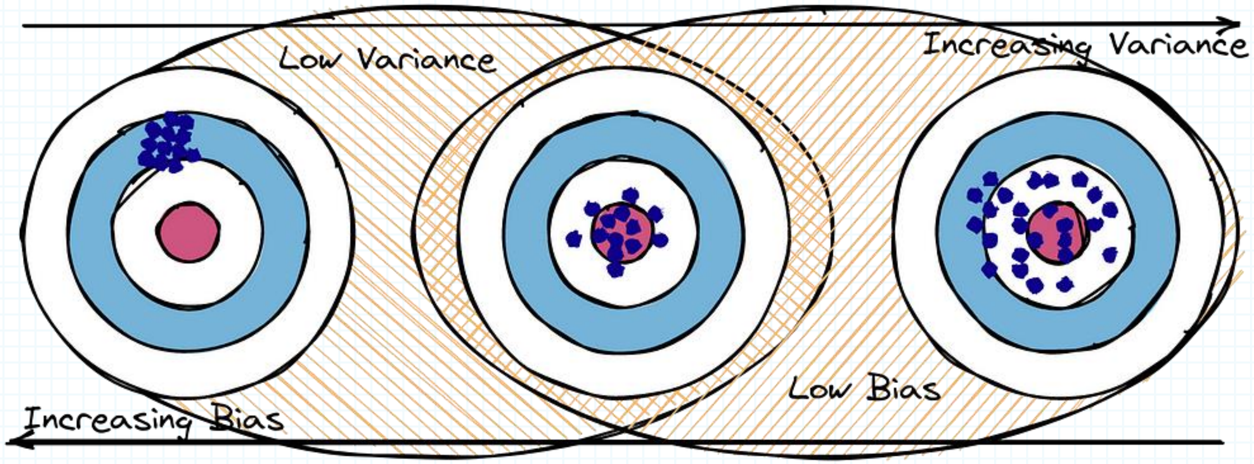
- Variance refers to **the changes in the model** when using <u>different</u> portions of the <u>training</u> data set.
- Simply stated, variance is the **variability in the model prediction**
- Different samples of training data yield different model fits

Bias-Variance Trade off

What does tons of training data do for us in terms of bias-variance trade off?

- ✓ Doesn’t affect bias
- ✓ Reduces variance
- ✓ Allows more complex model to be used [shift of optimal complexity to the right]





Bias-Variance Trade off

Training set Error	1%	15%	15%	0.5%
Testing set Error	11%	16%	30%	1%
Bias				
Variance				

DOG vs. CAT

Bias-Variance Trade off

Training set Error	1%	15%	15%	0.5%
Testing set Error	11%	16%	30%	1%
Bias	Low	High	High	Low
Variance	High	Low	High	Low

DOG vs. CAT

Error = Irreducible Error + Bias + Variance

Solutions!

- **High Bias:**
  - Bigger Network?
  - Train Longer?
- **High Variance:**
  - More Data
  - Regularization.

$$Err(x) = E\left[(Y - \hat{f}(x))^2\right]$$

$$Err(x) = \left(E[\hat{f}(x)] - f(x)\right)^2 + E\left[\left(\hat{f}(x) - E[\hat{f}(x)]\right)^2\right] + \sigma_e^2$$

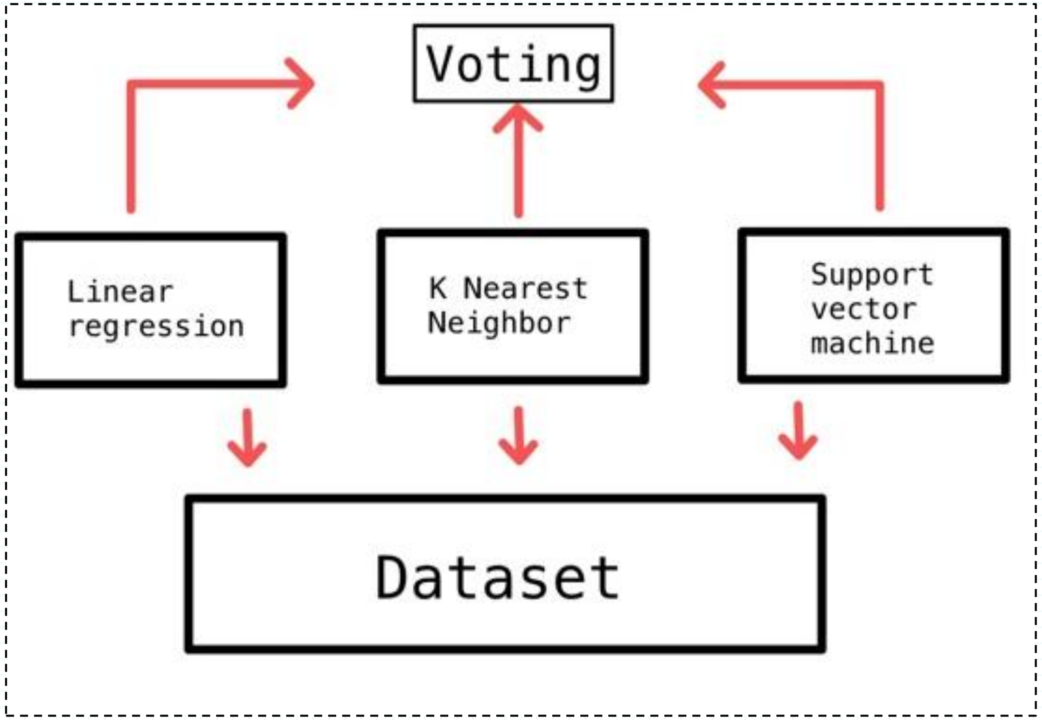
$$Err(x) = \mathrm{Bias}^2 + \mathrm{Variance} + \mathrm{Irreducible\ Error}$$

Practical Concerns for Machine Learning 2

Ensemble Methods

Bias-Variance Tradeoff & How Ensembles Help

- The goal in ML is to balance **bias** (error due to assumptions) and **variance** (error due to model sensitivity to data) to improve generalization.
- **Single** machine learning models **can fail** due to a balance issue between bias and variance.
- Ex: Basic Ensemble method



Ensemble Learning

- Ensemble Learning:

  Method that combines a collection of base learners to obtain performance improvements over its components

- Where do Learners come from?
  - Different learning algorithms
  - Algorithms with different choice for parameters
  - Data set with different features
  - Data set = different subsets
  - Different sub-tasks

Main Families of Ensemble Methods
Averaging Methods

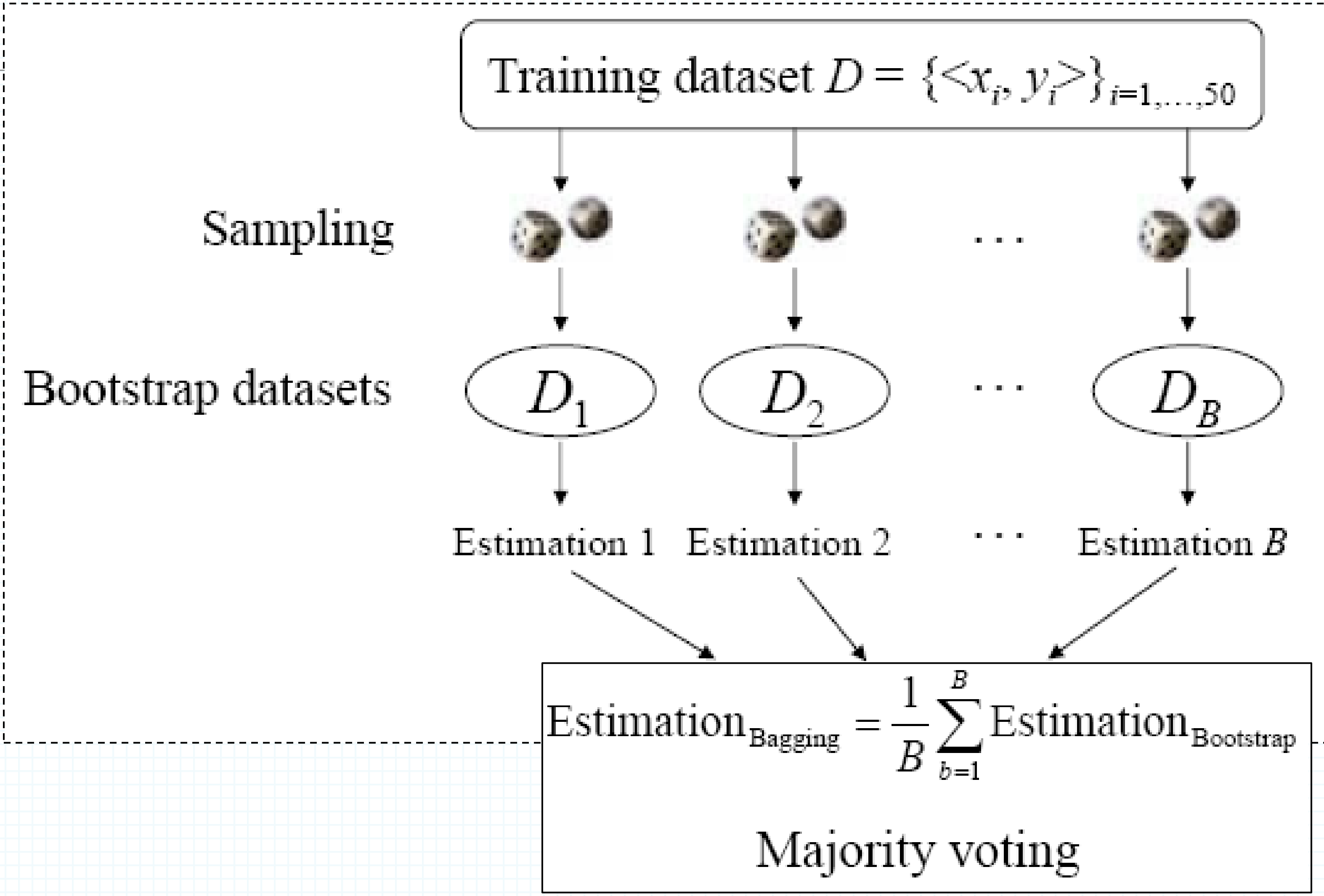
Main Families of Ensemble Methods
Averaging Methods

- Build multiple base estimators **independently** (Example: Random Forests).
- Combine their predictions using an **average** (for regression) or **majority vote** (for classification).
- Helps to **reduce variance** and improve stability.
- *Key Idea*: "Many weak, noisy models can create a strong, stable one when combined."
- Commonly-used ensemble methods:
  - Bagging (Bootstrap Aggregating): multiple models on random subsets of data samples
  - Random Subspace Method: multiple models on random subsets of features

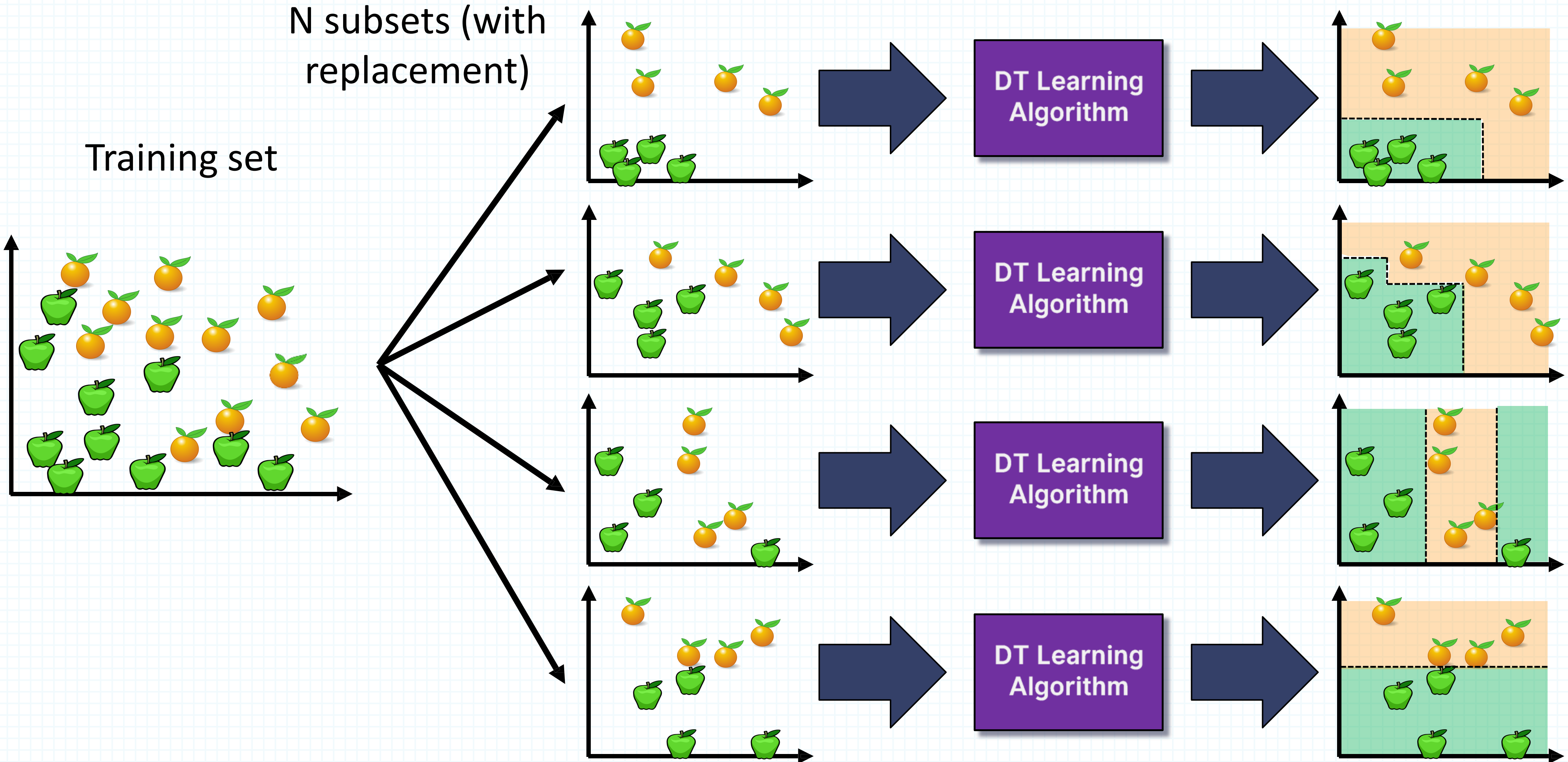
Helps when dealing with high-dimensional data

Bagging

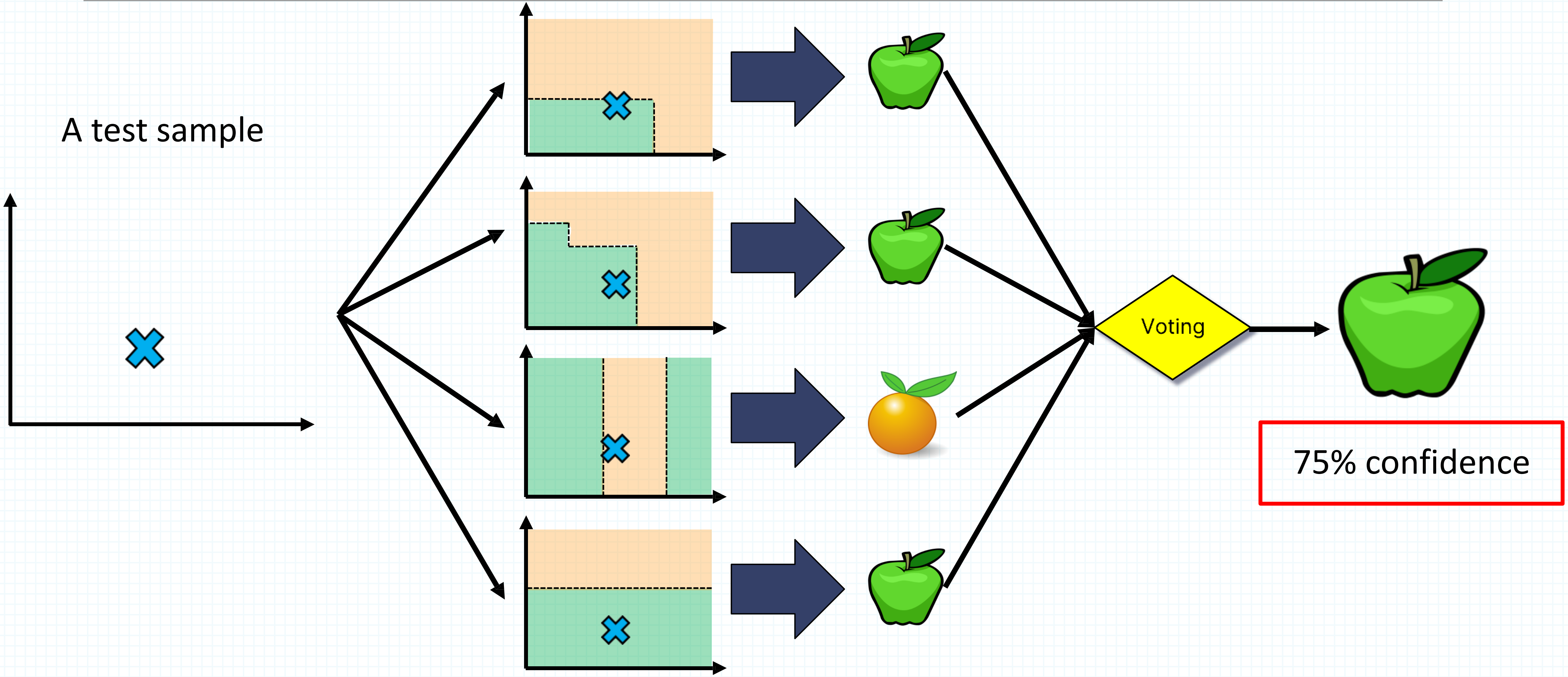
- In bagging, a random sample of data in a training set is selected with replacement—meaning that the individual data points can be chosen more than once
- Ensemble learning method that is commonly used to reduce **variance** within a noisy dataset (Why?)



Bagging (Training Stage)



Bagging (inference)



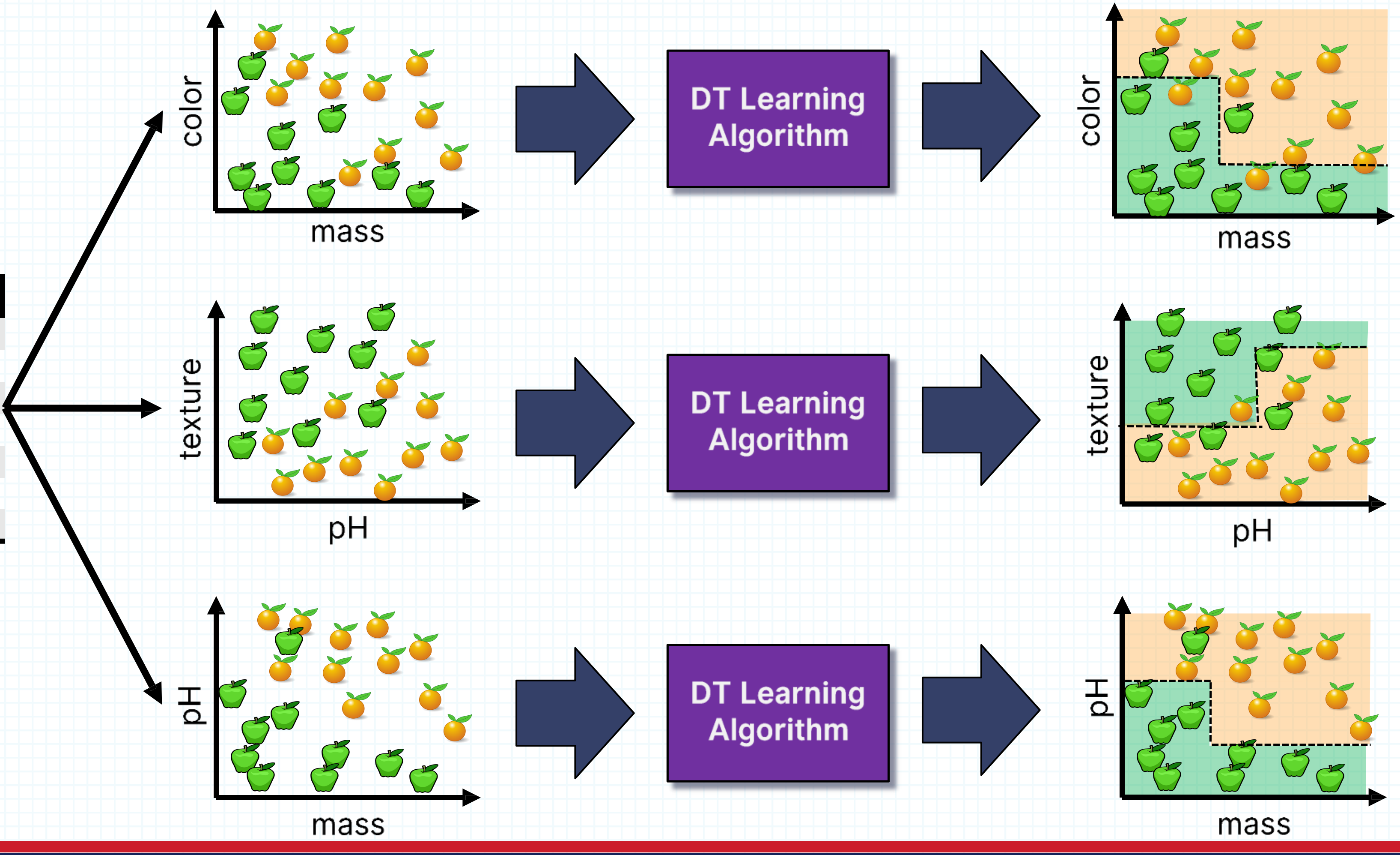
Random Subspace Method

- The principle is to increase diversity between members of the ensemble by restricting classifiers to work on different random subsets of the full feature space.
- Each classifier learns with a subset of size n, chosen uniformly at random from the full set of size N. Empirical studies have suggested good results can be obtained with the rule-of-thumb to choose $n = N/2$ features.

Random Subspace Method (Training)

Training data

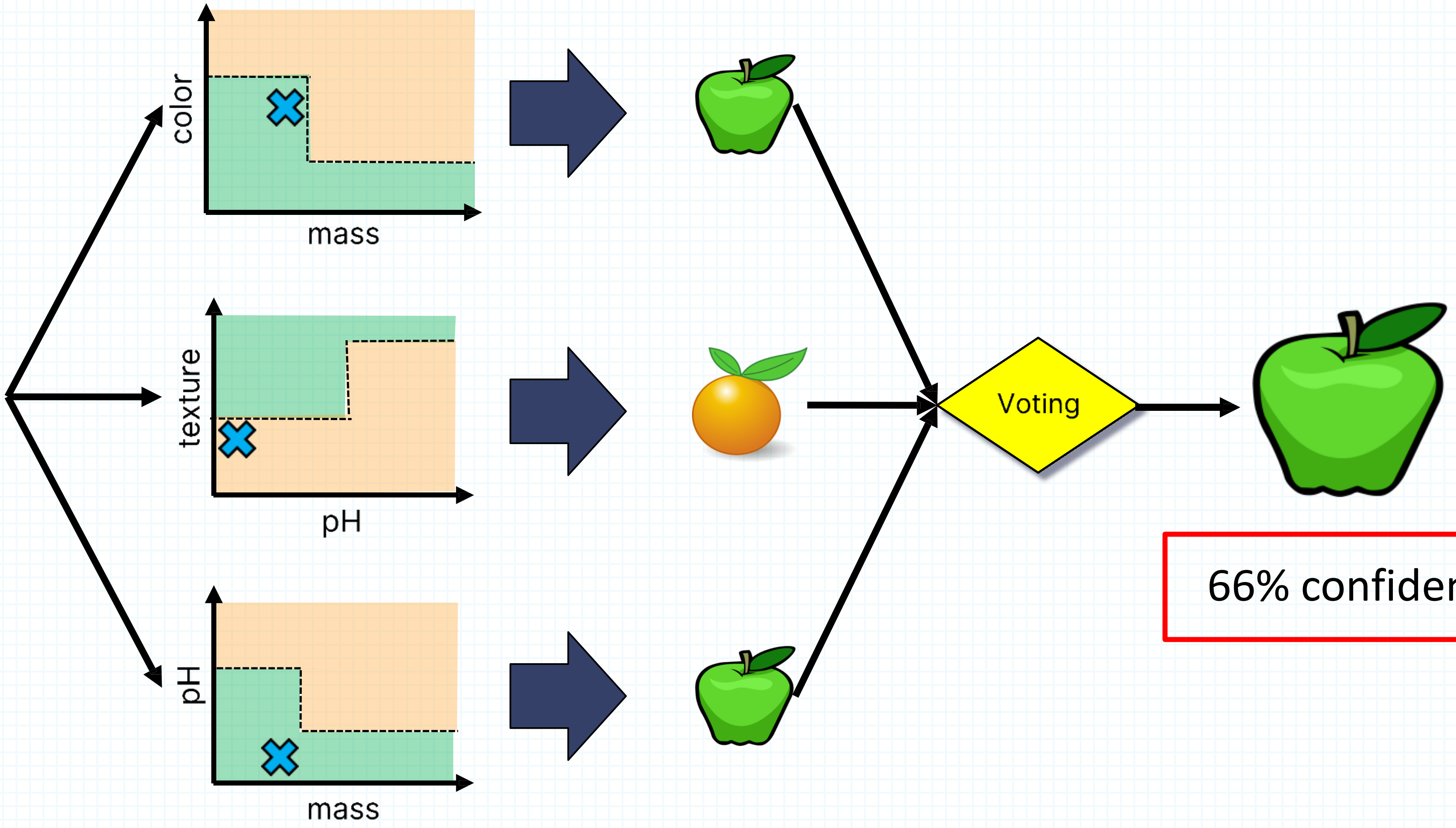
Mass (g)	Color	Texture	pH	Label
84	Green	Smooth	3.5	**Apple**
121	Orange	Rough	3.9	**Orange**
85	Red	Smooth	3.3	**Apple**
101	Orange	Smooth	3.7	**Orange**
111	Green	Rough	3.5	**Apple**
...				
117	Red	Rough	3.4	**Orange**



Random Subspace Method (inference)

A test sample

87	Red	Smooth	3.1



66% confidence

Example: Random Forests

- Random Forests:

  Instead of building a single decision tree and use it to make predictions, build many slightly different trees and combine their predictions

- We have a single data set, so how do we obtain slightly different trees?

  - 1. Bagging (**B**ootstrap **Agg**regat**ing**):
    - Take random subsets of data points from the training set to create N smaller data sets
    - Fit a decision tree on each subset

  - 2. Random Subspace Method (also known as Feature Bagging):
    - Fit N different decision trees by constraining each one to operate on a random subset of features

Main Families of Ensemble Methods
Boosting Methods

Main Families of Ensemble Methods
Boosting Methods

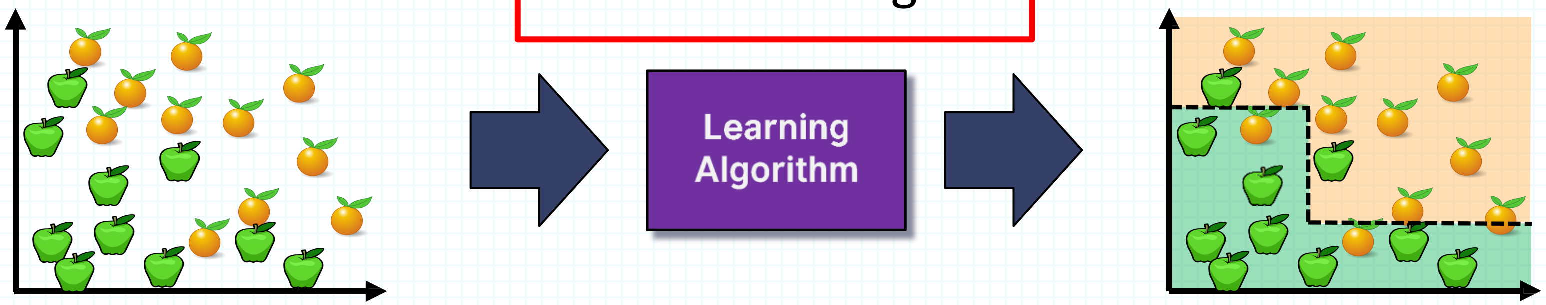
- Build base estimators **sequentially**, where each new model focuses on correcting the errors of the previous ones.
- Aims to **reduce bias** by creating a strong learner from multiple weak ones.
- Assigns more weight to hard-to-predict samples during training.
- *Key Idea*: “Turn a series of weak models into a strong one by learning from mistakes.”
- <u>Commonly-used ensemble methods:</u>
  - AdaBoost, Gradient Boosting, XGBoost

Boosting

- Boosting is an algorithm that **helps in reducing variance and bias in a machine learning ensemble**.
- The algorithm helps in the conversion of weak learners into strong learners by combining N number of learners.

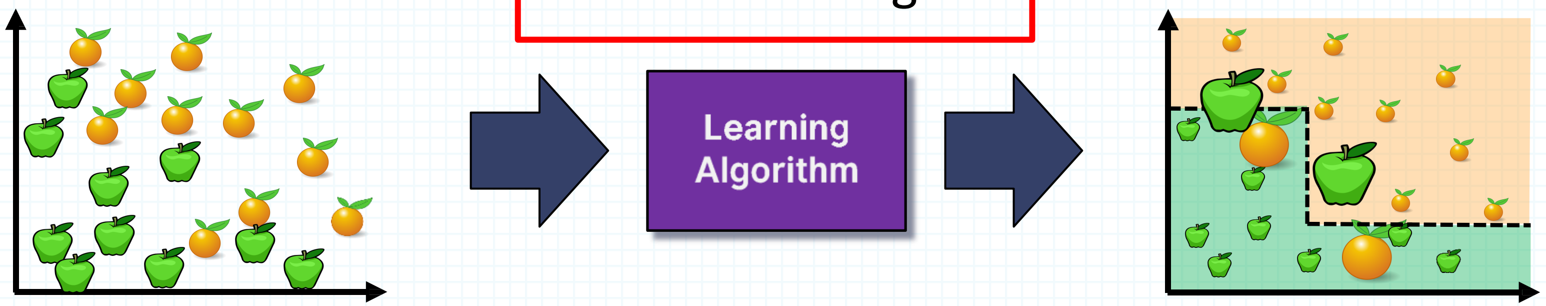
Boosting

All samples have the same weight



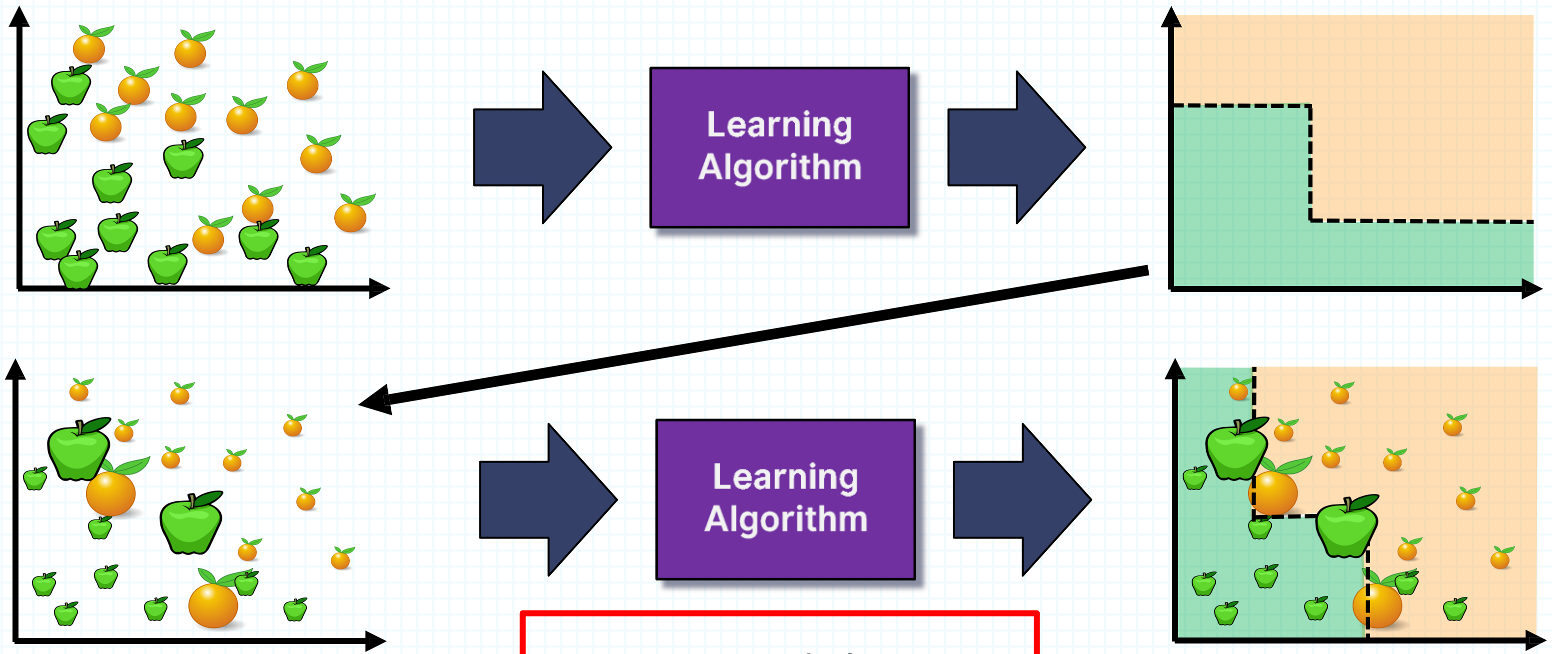
Boosting

All samples have the same weight



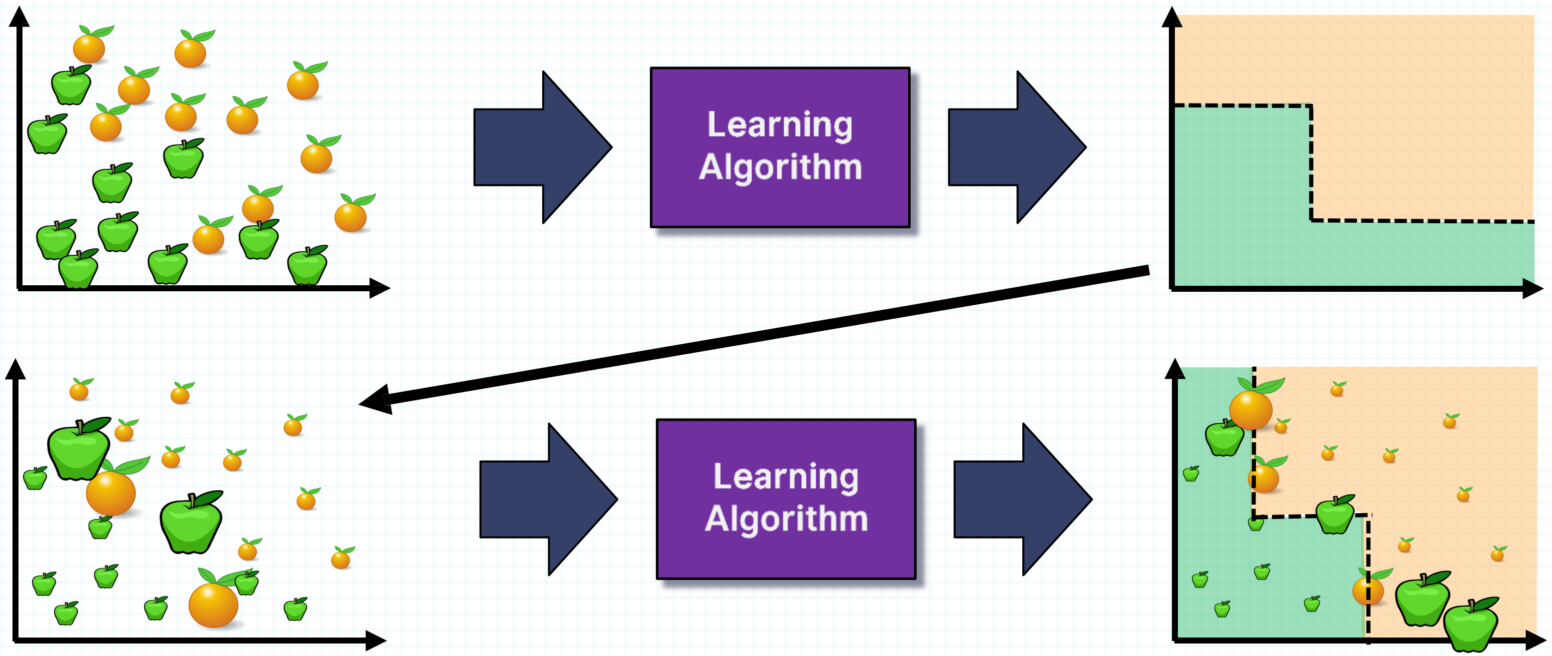
Reweight based on model's mistakes

Boosting



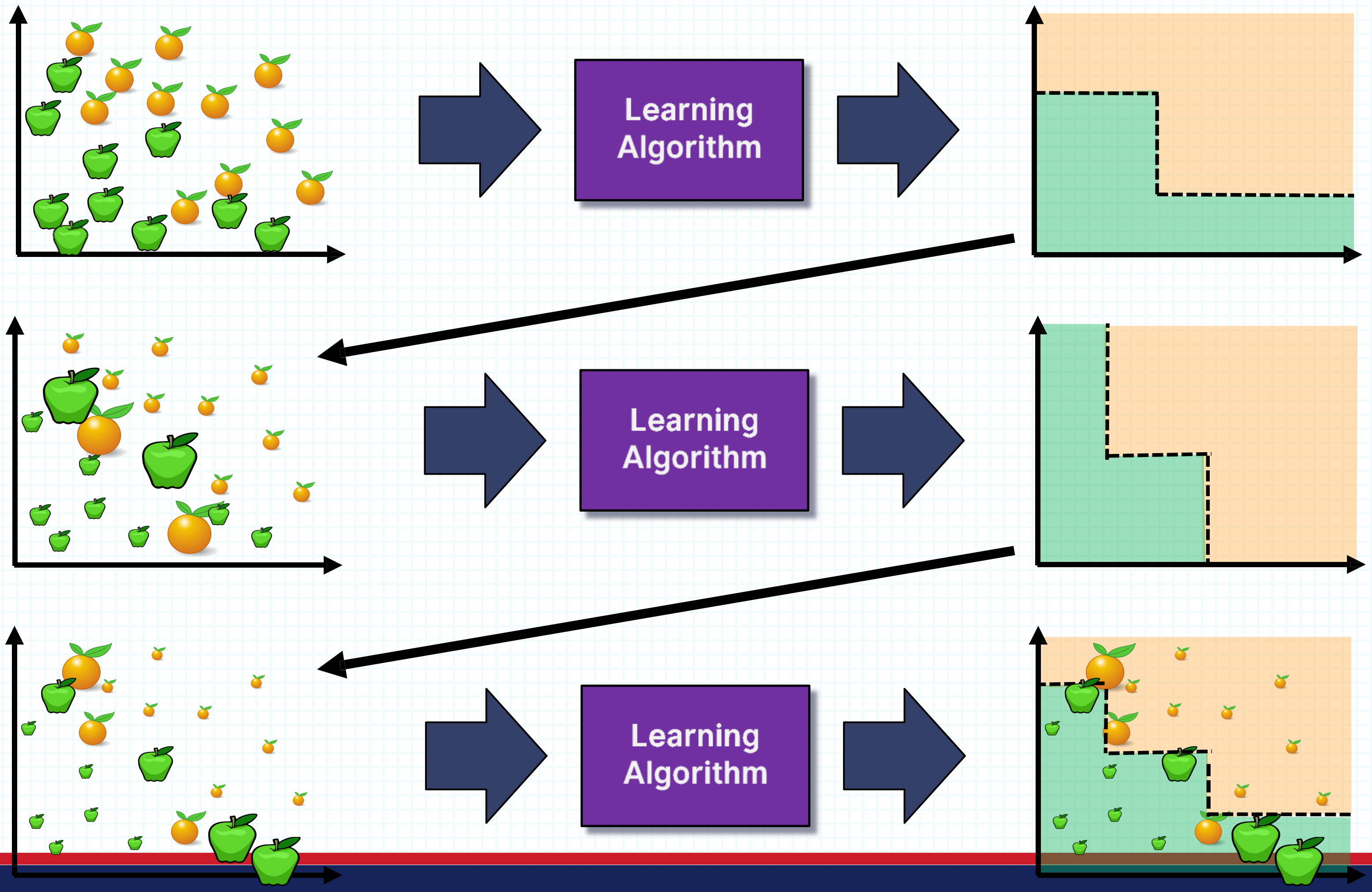
Next model sees weighted samples

Boosting

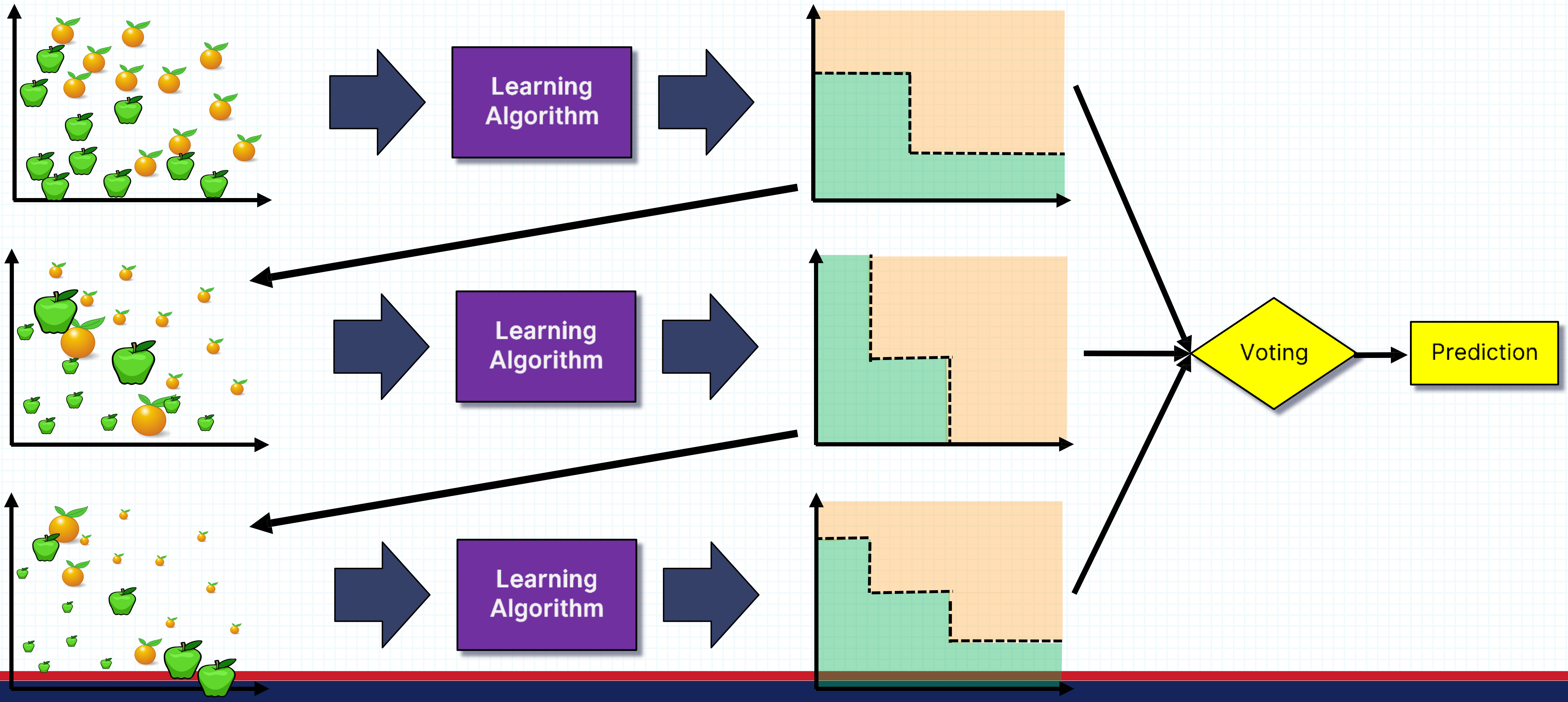


Reweight based on current model's mistakes

Boosting


Learning
Algorithm
Learning
Algorithm
Learning
Algorithm

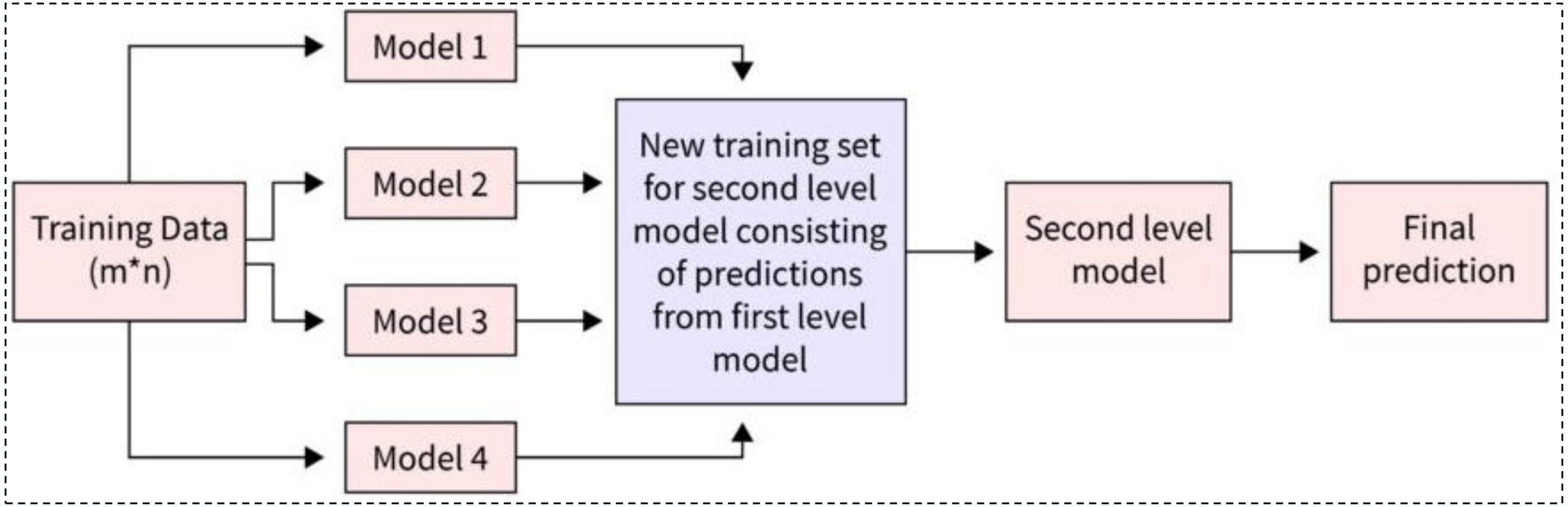
Boosting


Learning Algorithm
Learning Algorithm
Learning Algorithm
Voting
Prediction

Main Families of Ensemble Methods
Stacking Methods

Main Families of Ensemble Methods
Stacking Methods

- Combine multiple diverse models (level-0 learners) and train a **meta-model** (level-1) to make the final prediction.
- It works by taking the predictions from each model and feeding them into a final "meta-model" that learns how to best blend and stack their strengths.
- Can reduce both **bias and variance**.



Practical Concerns for Machine Learning 2

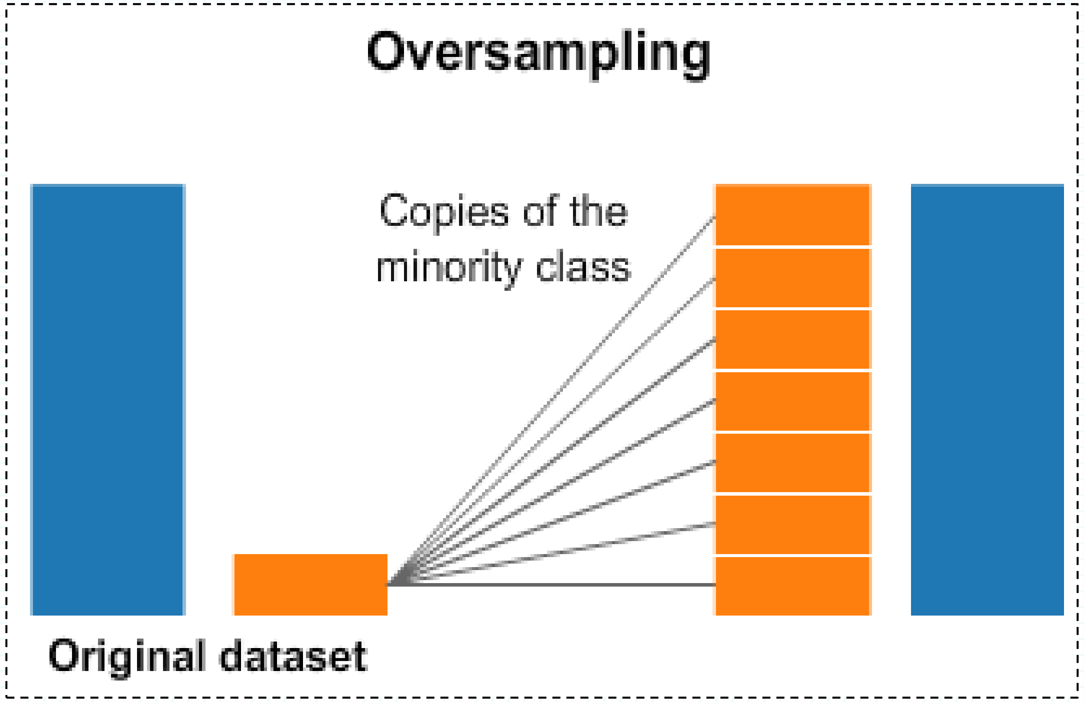
Imbalanced Dataset handling in Machine Learning

Introduction

- A dataset is said to be imbalanced when the distribution of target classes is highly skewed — i.e., one class (called the **majority** class) dominates the others (**minority** class/classes).
- **Example Scenarios:**
  - Fraud Detection: 99.8% non-fraudulent, 0.2% fraudulent transactions.
  - Medical Diagnosis: 95% healthy patients, 5% disease cases.
  - Manufacturing Fault Detection: 98% normal products, 2% defective.
- **Why is it a Problem?**
  - **Biased Models:** Standard classifiers tend to favor the majority class.
  - **Poor Generalization:** Minority classes are poorly learned → bad predictions where you need them most.
  - **Misleading Accuracy:** A classifier could reach 99% accuracy by predicting only the majority class.

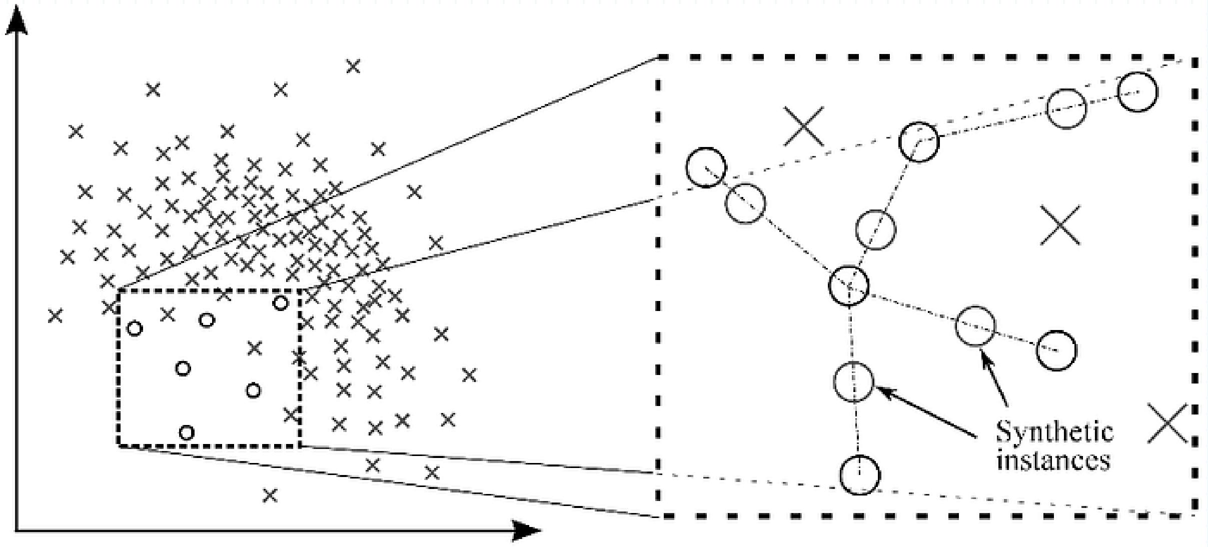
Over-Sampling Techniques

- **Random Over-Sampling (ROS):**
  - Duplicate minority class examples.
  - **Pros:** Simple, reduces imbalance.
  - **Cons:** Overfitting risk — models may memorize duplicated samples.



Over-Sampling Techniques

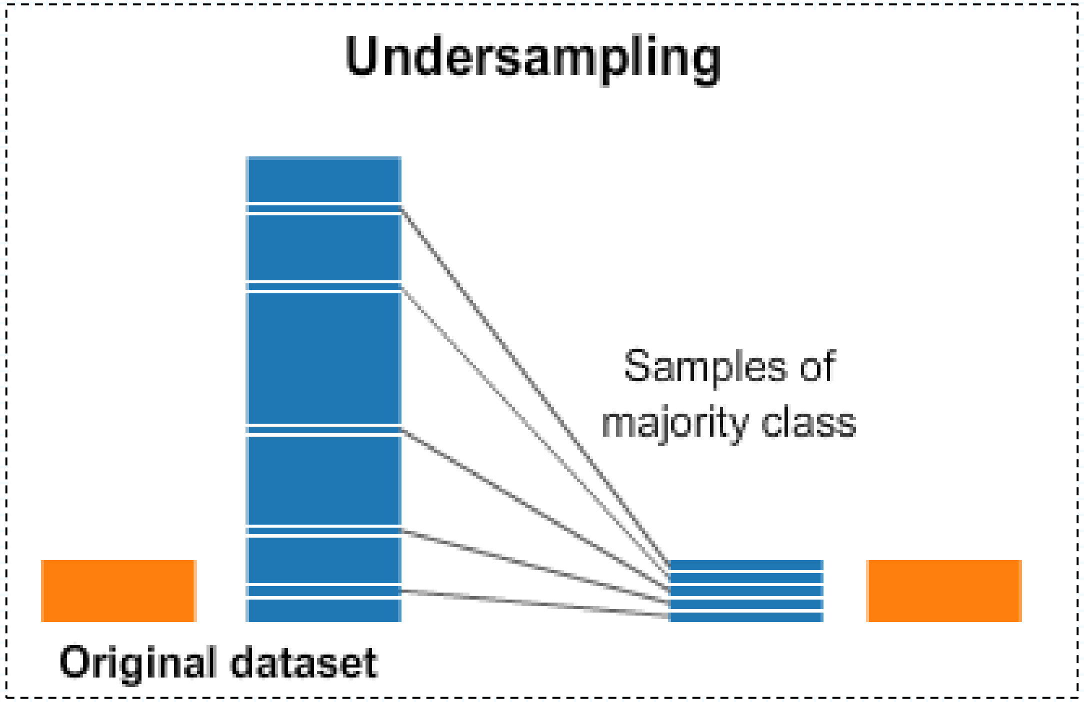
- **SMOTE (Synthetic Minority Over-sampling Technique):**
  - Generates **synthetic minority class samples** by interpolation between existing ones.
  - Avoids overfitting seen in simple duplication.
  - **Steps:**
    - Select random minority example.
    - Find its k-nearest minority neighbors.
    - Pick one randomly.
    - Generate new sample along the line connecting them.



  - **Limitations:**
    - Can generate **noisy or unrealistic samples** if minority class is spread widely.

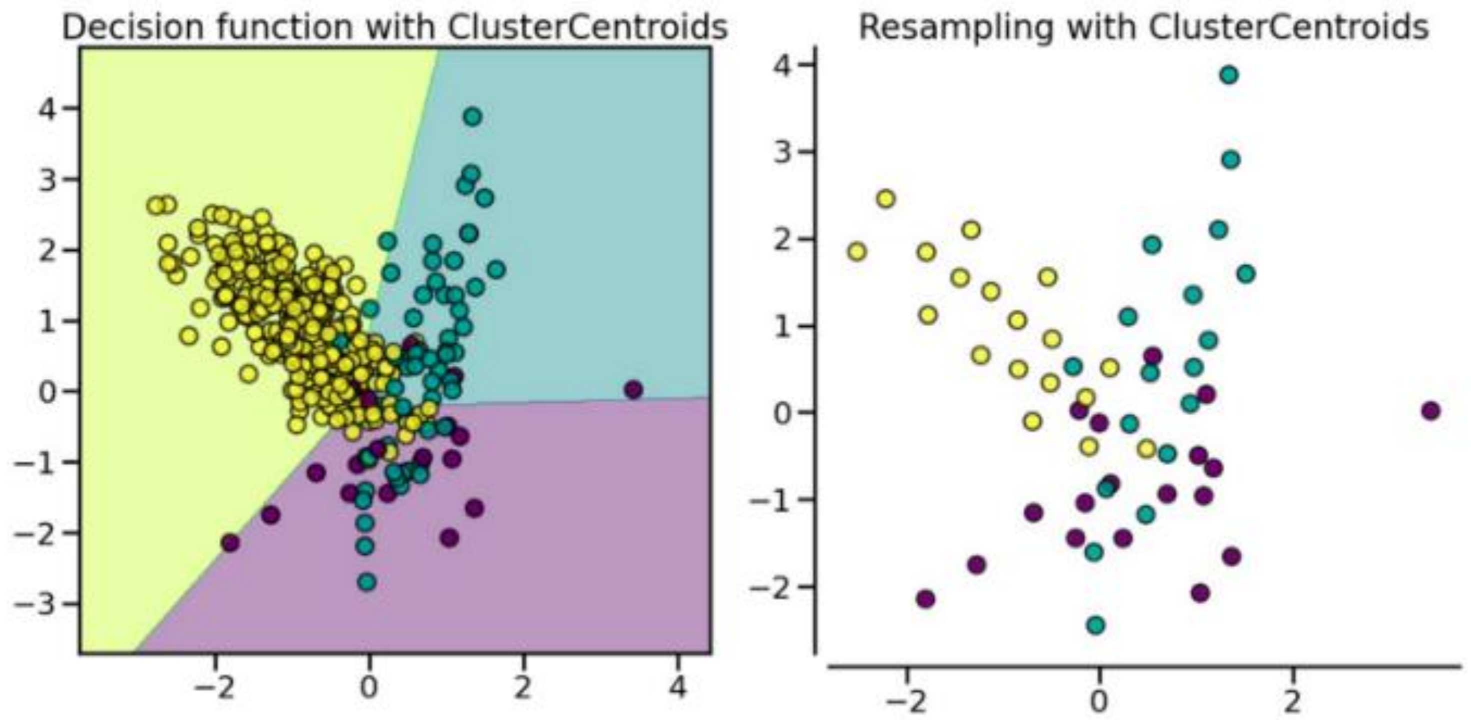
Under-Sampling Techniques

- **Random Under-Sampling (RUS):**
  - Removes random majority class samples.
  - **Pros:** Quick, reduces data size.
  - **Cons:** Potentially discards valuable information.



Under-Sampling Techniques

- **Cluster Centroids:**
  - Majority class samples are replaced by their cluster centroids.



Algorithm-Level Solutions

- **Cost-Sensitive Learning:**
  - Modify the **loss function** to penalize mistakes on minority class more heavily.
  - Example: In scikit-learn, many models (e.g., LogisticRegression, SVM, DecisionTree) accept **class_weight='balanced'**.

Actual / Predicted	Positive (Disease)	Negative (No Disease)
Positive (Disease)	0 (Correct)	10 (False Negative - Risky)
Negative (No Disease)	5 (False Positive - Unnecessary tests)	0 (Correct)