ARTIFICIAL INTELLIGENCE NEEDS REAL INTELLIGENCE

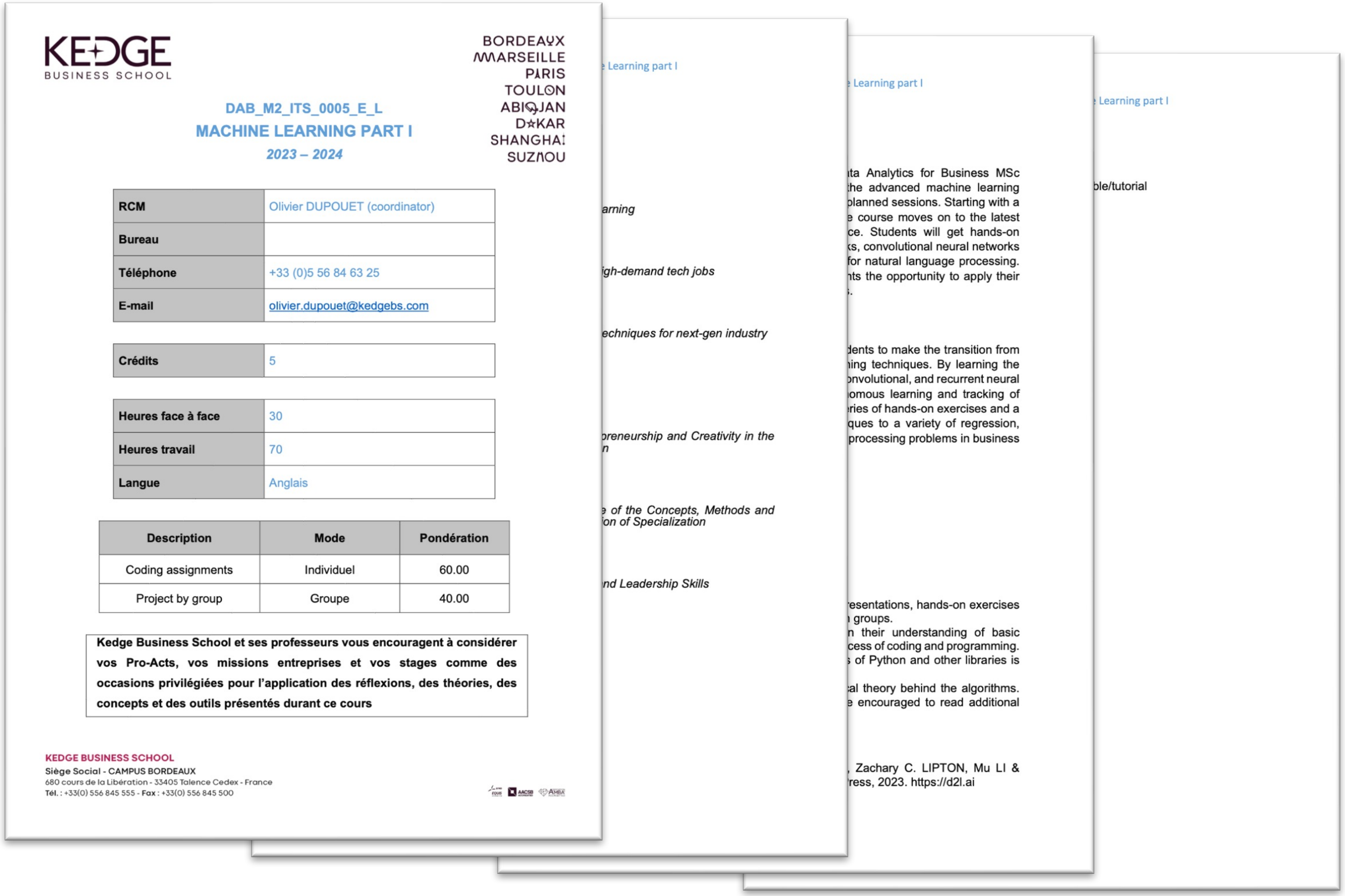
MSc DAB
Machine Learning
Part II

Professor: Tianyuan ZHANG
tianyuan.zhang@kedgebs.com

KEDGE
BUSINESS SCHOOL

Syllabus

- Available on **Learn**



KEDGE BUSINESS SCHOOL

BORDEAUX
MARSEILLE
PARIS
TOULON
ABIDJAN
DAKAR
SHANGHAI
SUZHOU

DAB_M2_ITS_0005_E_L
MACHINE LEARNING PART I
2023 – 2024

RCM	Olivier DUPOUET (coordinator)
Bureau	
Téléphone	+33 (0)5 56 84 63 25
E-mail	olivier.dupouet@kedgebs.com

Crédits	5

Heures face à face	30
Heures travail	70
Langue	Anglais

Description	Mode	Pondération
Coding assignments	Individuel	60.00
Project by group	Groupe	40.00

Kedge Business School et ses professeurs vous encouragent à considérer vos Pro-Acts, vos missions entreprises et vos stages comme des occasions privilégiées pour l'application des réflexions, des théories, des concepts et des outils présentés durant ce cours

KEDGE BUSINESS SCHOOL
Siège Social - CAMPUS BORDEAUX
680 cours de la Libération - 33405 Talence Cedex - France
Tél. : +33(0) 556 845 555 - Fax : +33(0) 556 845 500

Learning Objectives

- ***Ensemble learning***
- Transition from classical machine learning to ***deep learning***
    - Linear neural networks
        - Regression & Classification
    - Convolutional neural networks
        - Computer vision
    - Recurrent neural networks
        - Natural language processing
- Gain a foundation for ***self-study*** of future advances in AI

Pedagogy Methods

- This course won’t focus on the mathematical theory behind algorithms.
- For each session:
  - Lecture to present key topics and algorithms
  - Hands-on exercises with Jupyter Notebook

Pedagogy Methods

- Programming language
- Python library

Pedagogy Methods

- For debug:
  - Read **official documentations**
  - Read **official tutorials**

  - Search on stackoverflow, etc.
  - Ask ChatGPT, Google Gemini, Le Chat Mistral, etc.

Evaluation

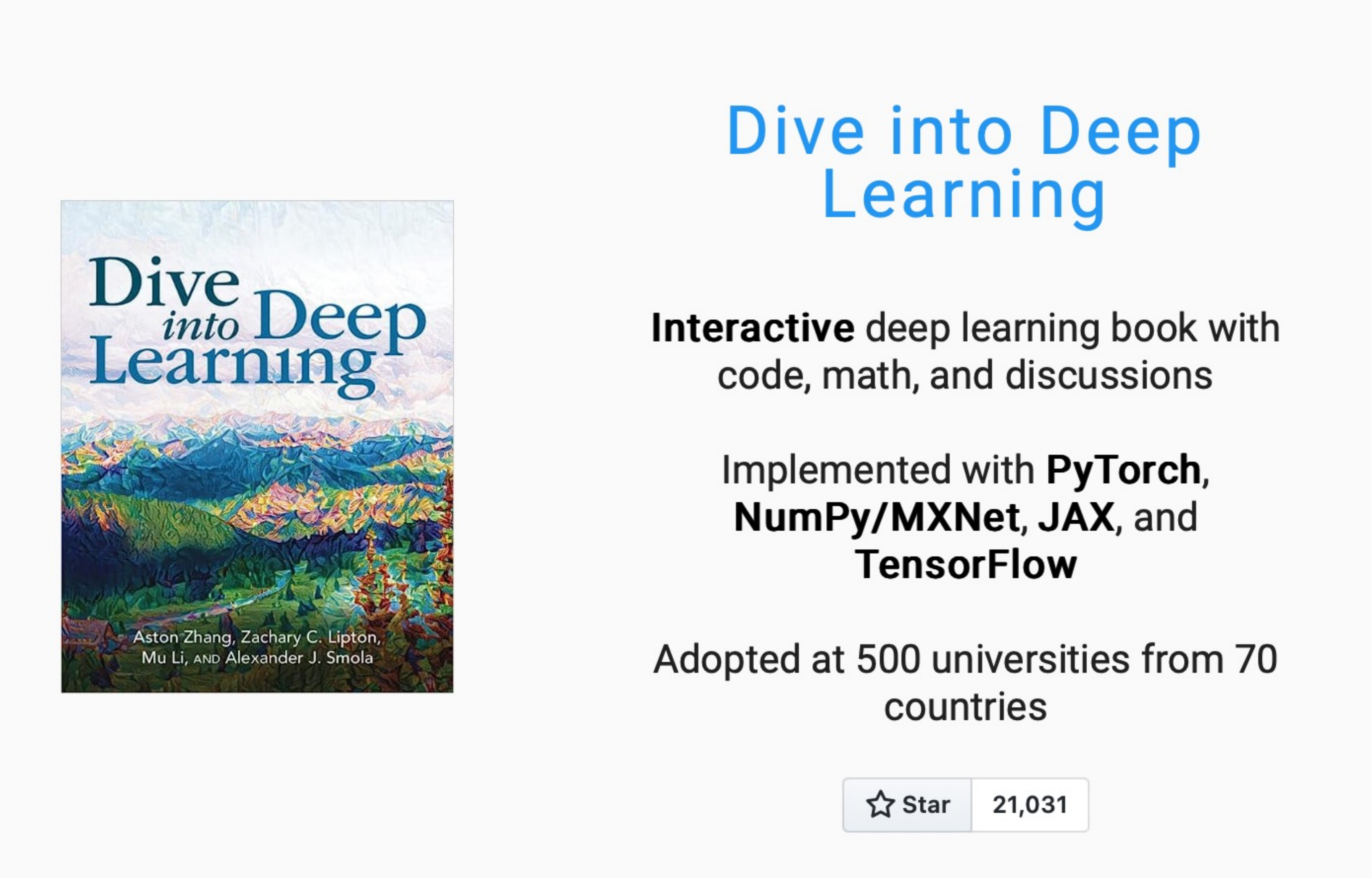
- Coding assignments → By individual → 60%
- Project → By group → 40%

Evaluation

MON	*TUE*	*WED*	THU	FRI	SAT	SUN
11/03	*12/03*	13/03	14/03	15/03	16/03	17/03
18/03	*19/03*	*20/03*	21/03	22/03	23/03	24/03
25/03	*26/03*	*27/03*	28/03	29/03	30/03	31/03
08/04	*09/04*	*10/04*	11/04	12/04	13/04	14/04
15/04	*16/04*	*17/04*	18/04	19/04	20/04	21/04
22/04	23/04	*24/04*	25/04	26/04	27/04	28/04

Supplemental Materials

- https://d2l.ai



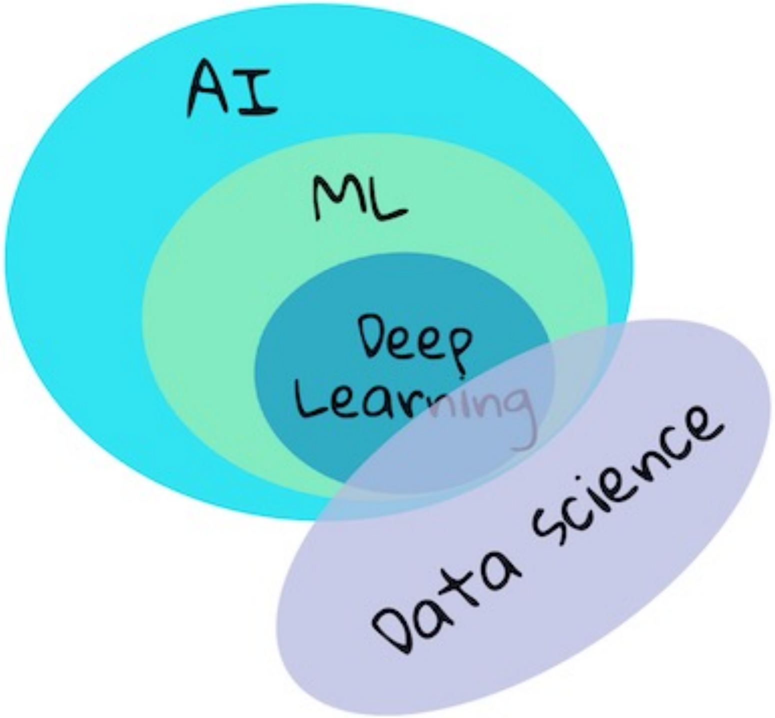
ARTIFICIAL INTELLIGENCE NEEDS REAL INTELLIGENCE

Recap of Classical Machine Learning

Professor: Tianyuan ZHANG
tianyuan.zhang@kedgebs.com

AI & ML & DL & DS

- **Artificial Intelligence (AI)**
  - A science of getting machines to accomplish tasks that typically require human level intelligence.
- **Machine Learning (ML)**
  - A subset of AI, the most successful and popular approach to AI
  - Use specialized algorithms to make predictions by learning from data
- **Deep Learning (DL)**
  - A subset of ML that uses deep artificial neural networks for learning



KEDGE BUSINESS SCHOOL

Image source: https://github.com/microsoft/ML-For-Beginners/blob/main/1-Introduction/1-intro-to-ML/README.md

AI & ML & DL & DS

- **Machine Learning (ML)**
  - Learn from data
  - Create models to make predictions
  - Focus on the future
- **Deep Science (DS)**
  - Analyze and interpret data
  - Extract insights from data
  - Focus on the past

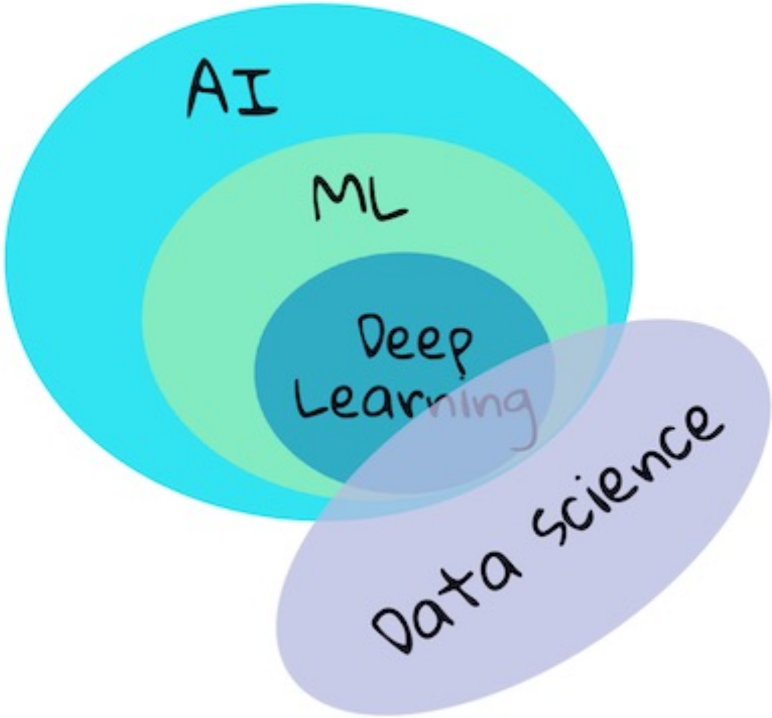
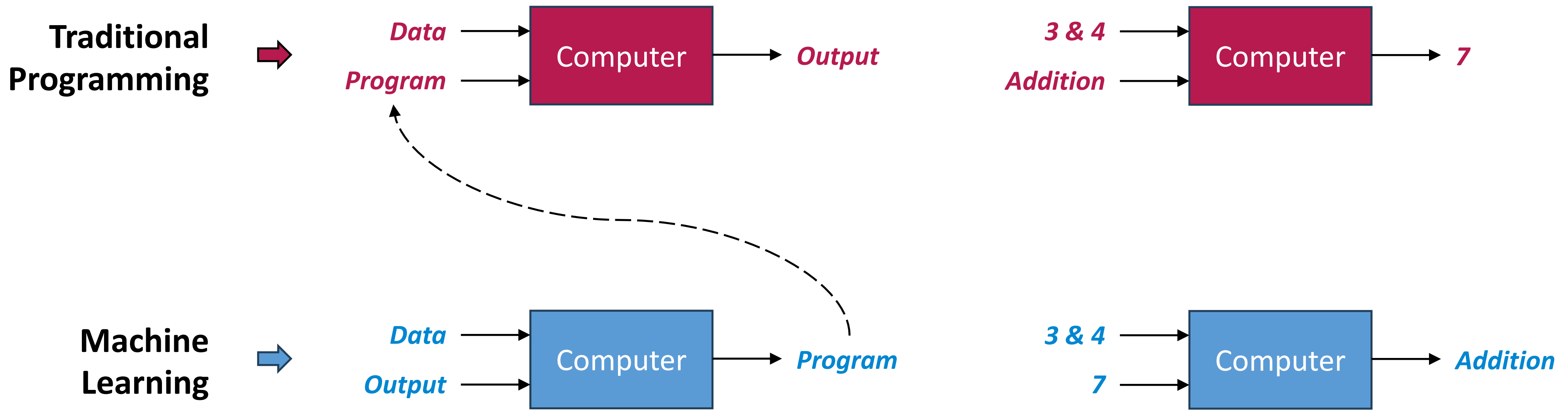


Image source: https://github.com/microsoft/ML-For-Beginners/blob/main/1-Introduction/1-intro-to-ML/README.md

Definition of Machine Learning

- Use specialized algorithms that can **learn from data** and **generalize to unseen data**, and thus perform tasks **without explicit instructions**.



Key Components of Machine Learning

- **Data**:
  - **Feature**
    - A feature is a measurable property of your data.
    - In ML, a feature is an **input** variable → x
    - A data set may contain multiple features → $x_1, x_2, \dots, x_N$
  - **Label**
    - A label is the **target** we try to predict.
    - In ML, a label is the **output** variable → y
  - **Example**
    - An example is a particular instance of data
      - **Labeled example** → $(x_1, x_2, \dots, x_N, y)$
      - **Unlabeled example** → $(x_1, x_2, \dots, x_N)$

Features *Label*

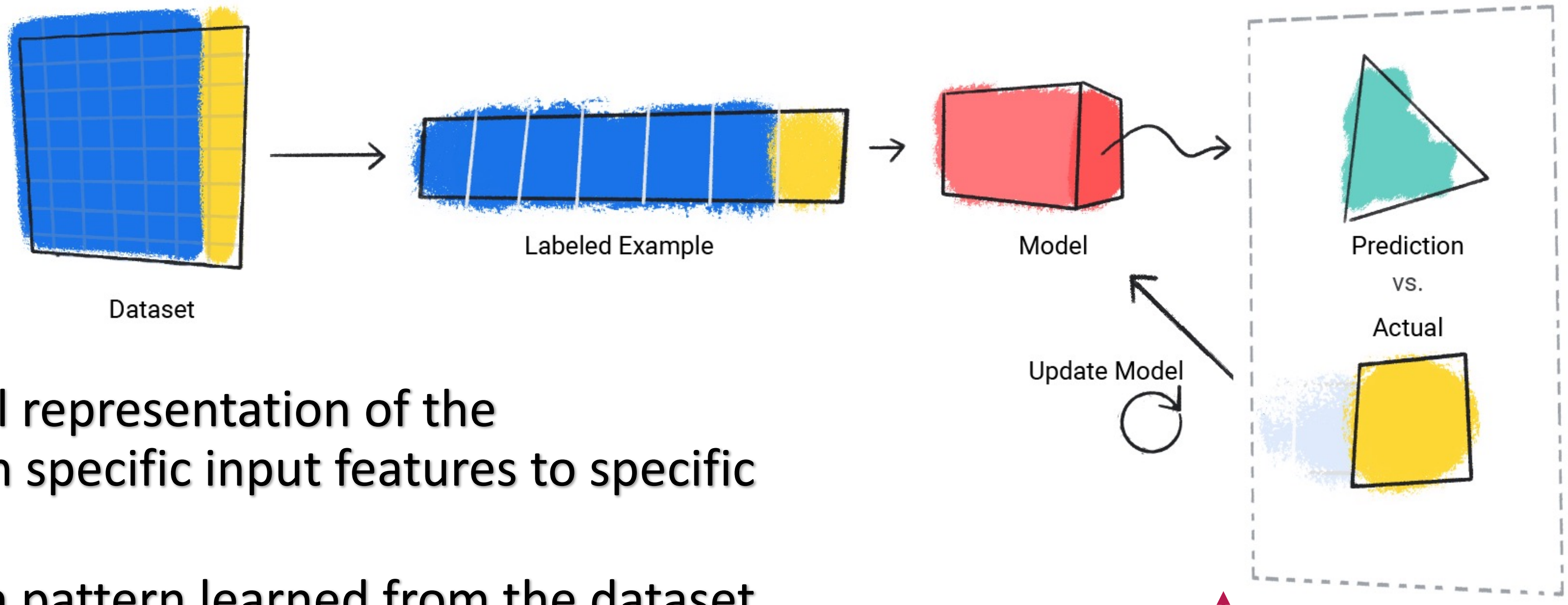
Long	Lat	City
2°21'E	48°51'N	Paris
0°7'E	51°31'N	London
13°23'E	52°31'N	Berlin
116°24'E	39°54'N	Beijing
151°12'E	33°51'S	?

Labeled Example → Berlin row

Unlabeled Example → ? row

Key Components of Machine Learning

- **Model**:
  - Build-time
    - Training
    - Evaluation
  - Inference
  - The mathematical representation of the relationships from specific input features to specific output labels.
  - Define the hidden pattern learned from the dataset, which can be used to make predictions on unseen data.



The typical training process of a model using supervised learning algorithms.

Image source: https://developers.google.com/machine-learning/intro-to-ml/supervised

Types of Machine Learning

- Machine learning approaches are traditionally divided into three broad categories, which corresponding to **learning paradigms.**

Supervised learning

Unsupervised learning

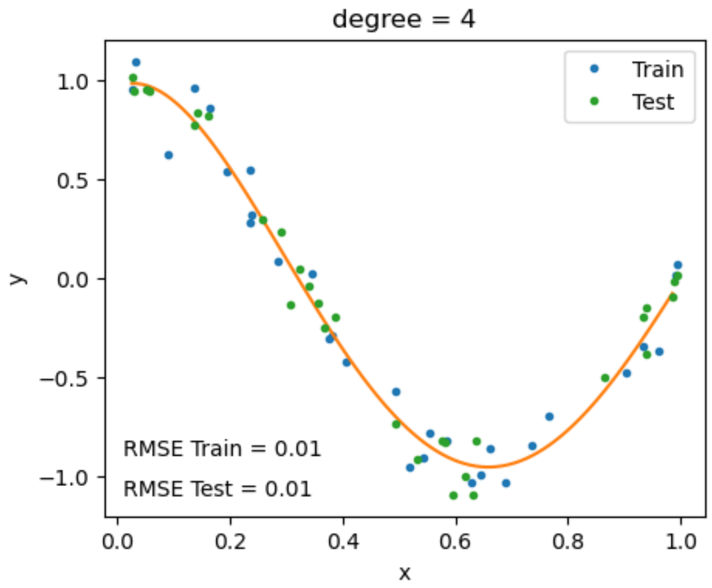
Reinforcement learning

Recap of Supervised Learning

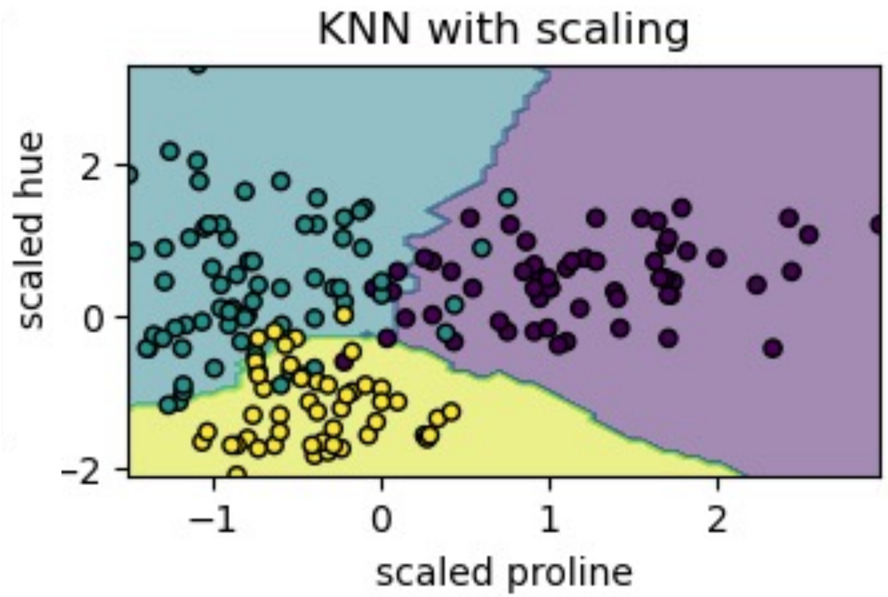
- **Supervised learning**
  - The algorithm is presented with **labeled examples**
  - The goal is to learn the hidden pattern between input features and output labels
  - After seeing lots of labeled examples, the model can make predictions on unseen unlabeled data
  - Two prediction tasks
    - **Regression**
      - Predict a continuous numeric value
    - **Classification**
      - Predict discrete categories or
      - Predict probabilities of the input example belongs to each category

Recap of Supervised Learning

- Regression
  - Estimate the relationship between features and labels by fitting to the training data



- Classification
  - Learn the way to divide the feature space into different parts, each represent a category



Recap of Supervised Learning

- Regression
  - Estimate the relationship between features and labels by fitting to the training data
  - **Algorithms**
    - Simple linear regression
    - Multiple linear regression
    - Polynomial regression

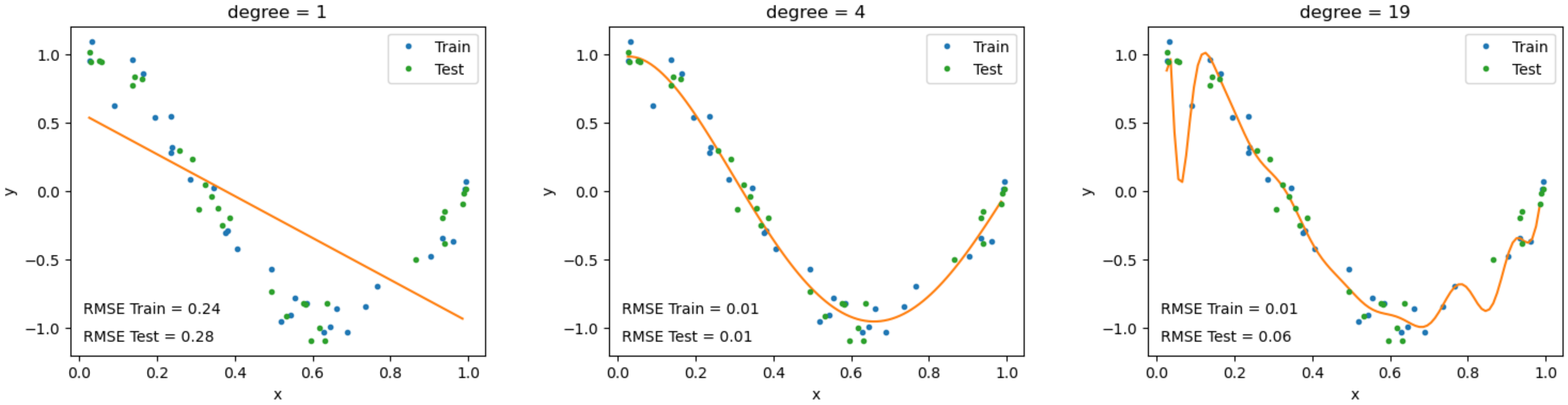
- Classification
  - Learn the way to divide the feature space into different parts, each represent a category
  - **Algorithms**
    - Logistic regression
    - K-Nearest neighbors
    - Support vector machine
    - Decision tree

Recap of Supervised Learning

- **Regression metric**
  - Measure the error of predictions
    - Residual
    - MSE
    - RMSE
    - MAE
    - R^2
- **Classification metric**
  - Measure the correctness of predictions
    - Accuracy
    - Confusion matrix
      - Precision
      - Recall
      - F1-score
    - ROC curve
    - Area under ROC curve

Recap of Supervised Learning

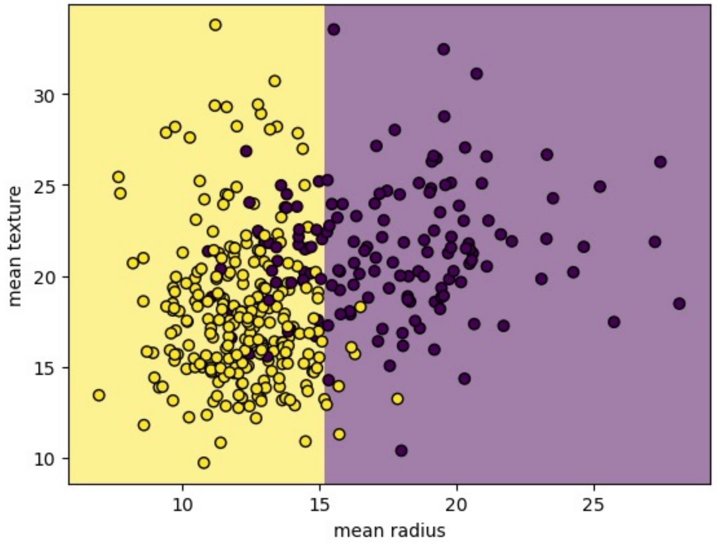
- **Under-fitting & Over-fitting**
  - Regression



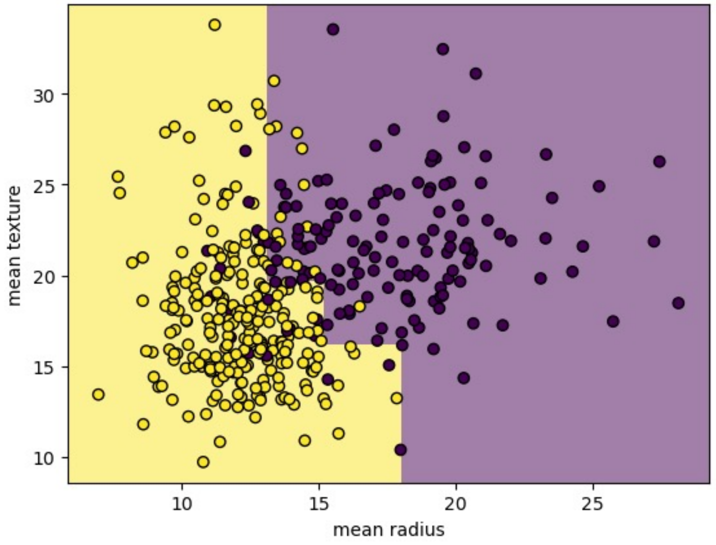
Under-fitting *Well-fitting* *Over-fitting*

Recap of Supervised Learning

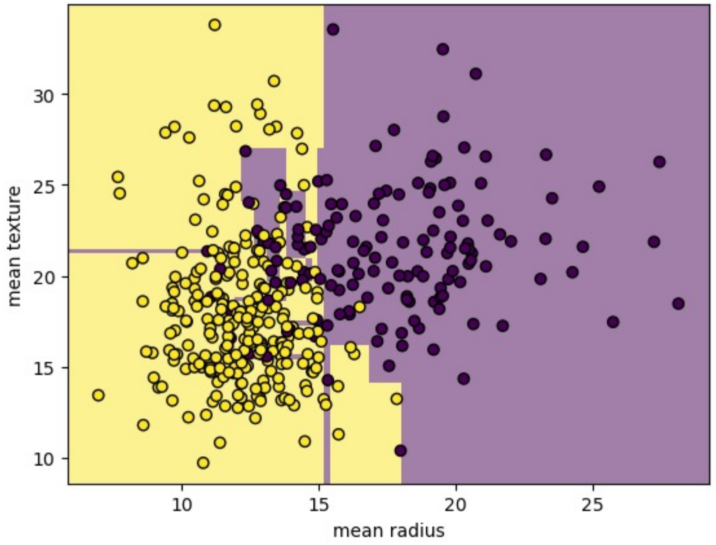
- **Under-fitting & Over-fitting**
  - Classification



Under-fitting



Well-fitting

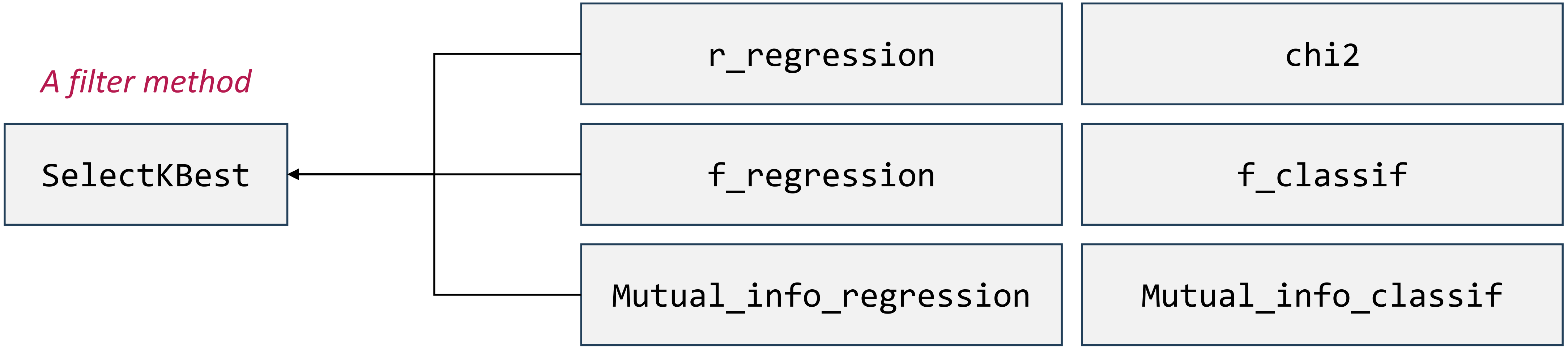


Over-fitting

Recap of Supervised Learning

- **Feature selection**
  - Select a subset of relevant features for model construction
  - Improve model performance, reduce model complexity, reduce training time

A filter method



Score function for regression task

Score function for classification task

Recap of Supervised Learning

- **Hyper-parameter**
  - Hyper-parameter controls model's training process
  - The value of hyper-parameter is defined before training
  - The value of hyper-parameter remains unchanged during training
- Hyper-parameter tuning
  - Search the best combination of values of hyper-parameters
  - **Grid search**
  - **Random search**
- Select the best combination based on the performance on **validation** set

Recap of Supervised Learning

- **Cross-validation**

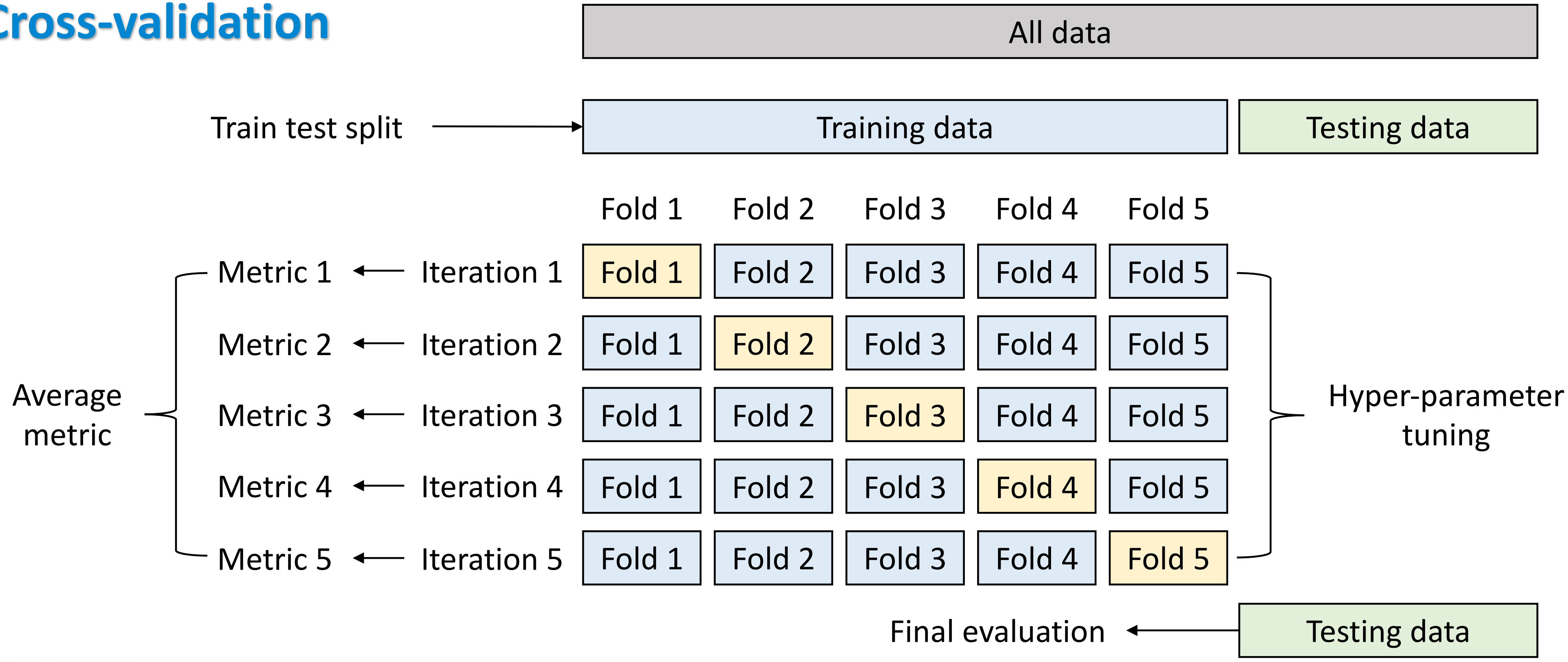


Image source: https://scikit-learn.org/stable/modules/cross_validation.html

Recap of Supervised Learning

- **Best practice**

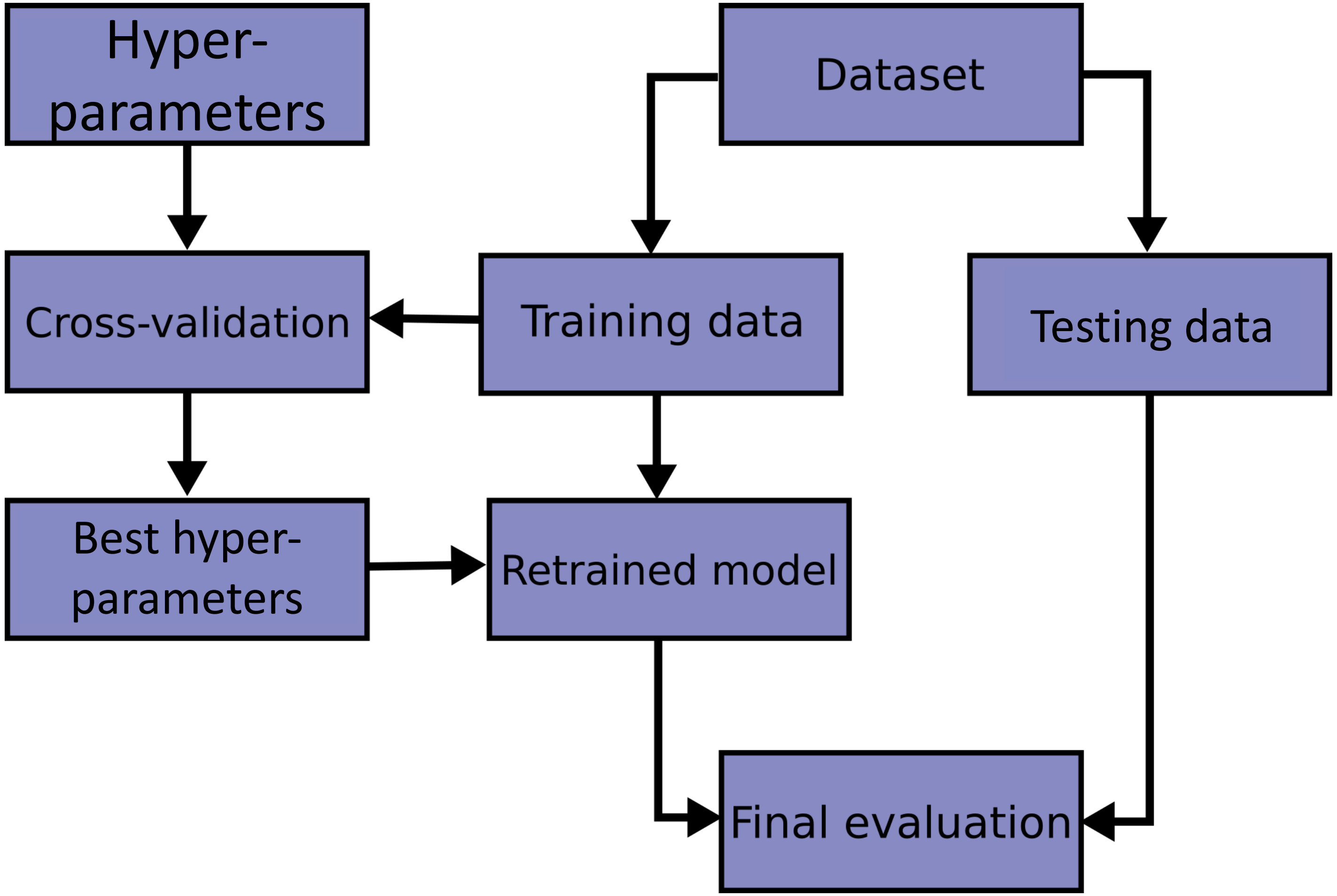


Image source: https://scikit-learn.org/stable/modules/cross_validation.html

Recap of Unsupervised Learning

- **Unsupervised Learning**
  - Learn **hidden patterns or structures** exclusively from **unlabeled data**
  - Three types of common unsupervised learning algorithms
    - **Clustering**:
      - Natural groups → Clusters
    - **Association rule mining**:
      - Frequent co-occurrence → Association rules
    - **Dimensionality reduction**:
      - Simplifications → Compressed data

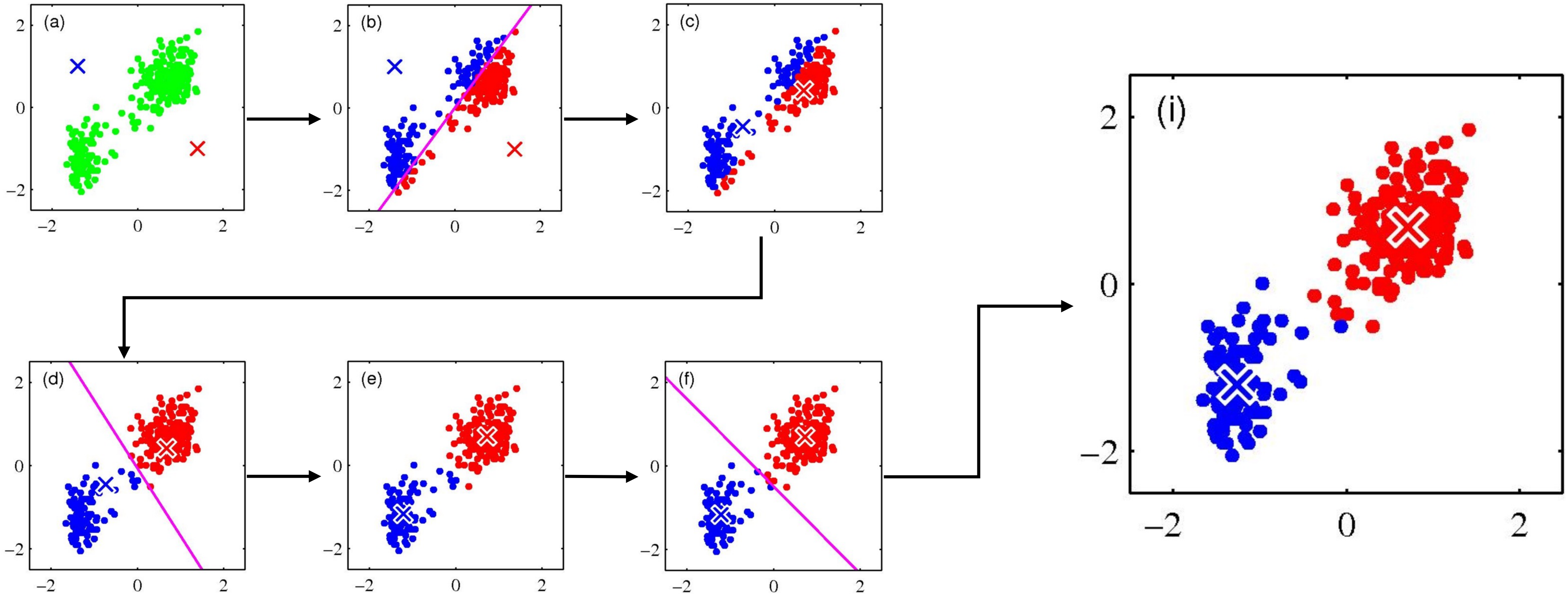
KEDGE BUSINESS SCHOOL

Recap of Unsupervised Learning

- **Clustering**
  - Grouping **unlabeled** data into different natural **clusters**
  - Data points within the same cluster are similar.
  - Data points in different clusters are different.
  - The clusters are not pre-defined by humans before clustering
  - We need to interpret the clustering results after clustering
- Two clustering algorithms
  - **K-Means clustering** → Centroid-based method
  - **Hierarchical clustering** → Connectivity-based method

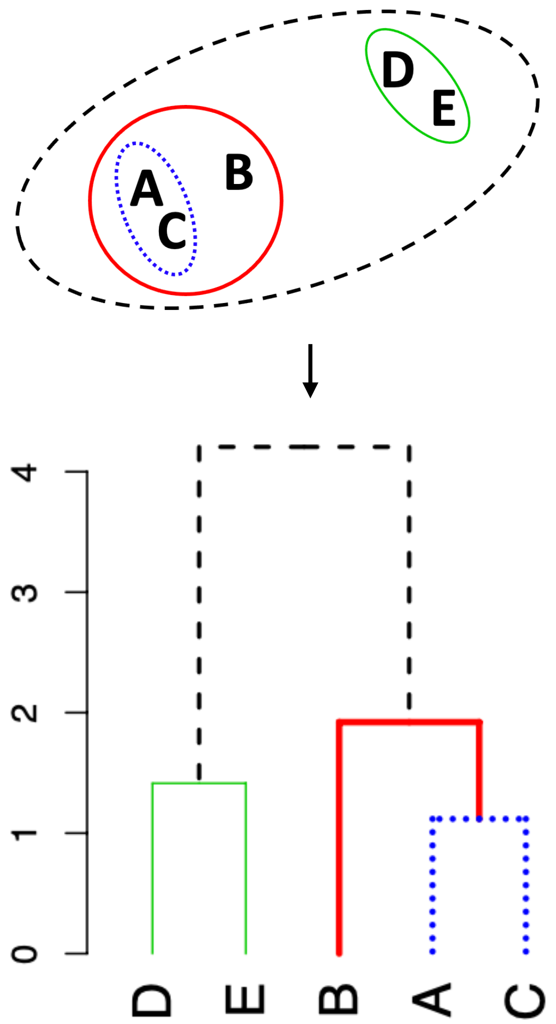
Recap of Unsupervised Learning

- **K-Means Clustering**



Recap of Unsupervised Learning

- **Hierarchical Clustering**
  - **Agglomerative clustering**
    - Bottom-up approach
    - Every point starts in its own cluster, merging clusters successively
  - Divisive clustering
    - Top-down approach
    - All point starts in one cluster, splitting clusters successively



Recap of Unsupervised Learning

- Association Rule Mining
  - Rule-based unsupervised learning algorithms
  - Identify frequent co-occurrence and generate association rules
  - Market Basket analysis
    - Item
      - Products sold in a supermarket
    - Basket
      - A set of items
      - The products one customer buys in one transaction

Transaction 1	🍎 🍺 🍚 🍗
Transaction 2	🍎 🍺 🍚
Transaction 3	🍎 🍺
Transaction 4	🍎 🍐
Transaction 5	🍼 🍺 🍚 🍗
Transaction 6	🍼 🍺 🍚
Transaction 7	🍼 🍺
Transaction 8	🍼 🍐

KEDGE BUSINESS SCHOOL

Image source: https://www.kdnuggets.com/2016/04/association-rules-apriori-algorithm-tutorial.html

Hands-on Exercise

- Exercise 01

ARTIFICIAL INTELLIGENCE NEEDS REAL INTELLIGENCE

Ensemble Learning I

Professor: Tianyuan ZHANG
tianyuan.zhang@kedgebs.com

KEDGE
BUSINESS SCHOOL

Outline

- **Ensemble Learning**
- Bagging
- Bias & Variance
- Random Forests

Ensemble Learning

- A machine learning paradigm where multiple models (often called "**weak learners**") are trained to solve the same problem and combined to get better results.
- Rationale
  - No single model can capture all the patterns in the data perfectly.
  - Leverage the strength and compensate for the weaknesses of individual models.
  - A group of "weak learners" can come together to form a "strong learner".
  - **Wisdom of the crowd**

Ensemble Learning

- **Wisdom of the crowd**
  - At a 1906 country fair in Plymouth, 800 people participated in a contest to estimate the weight of an ox.
  - Statistician Francis Galton observed that

The median guess

↓

*800 guesses: 450, 575, … , 1205, **1207**, 1225, … , 1600, 1750 pounds*

↓

The median guess is within 1% of the true weight.

*True weight: **1198** pounds*

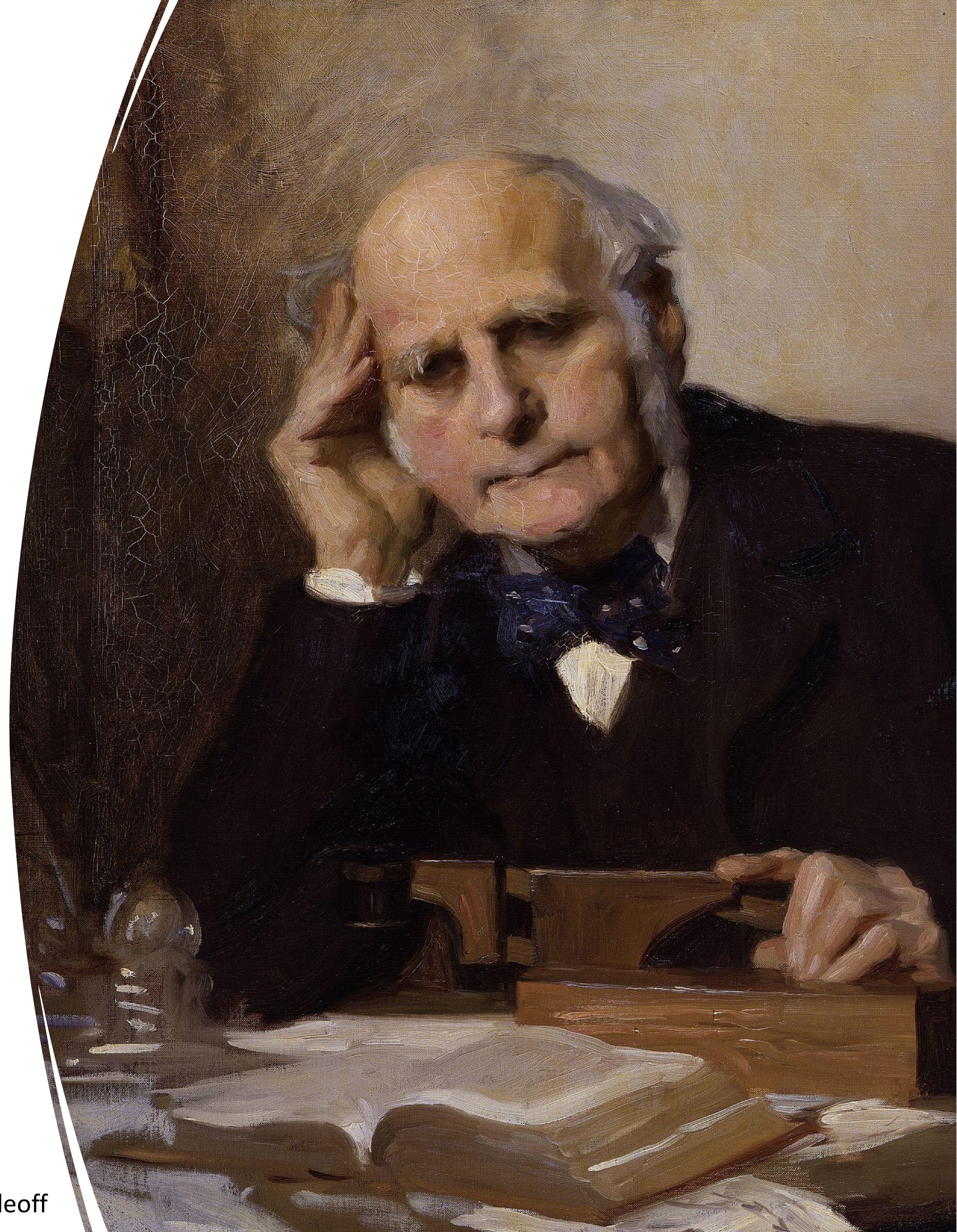
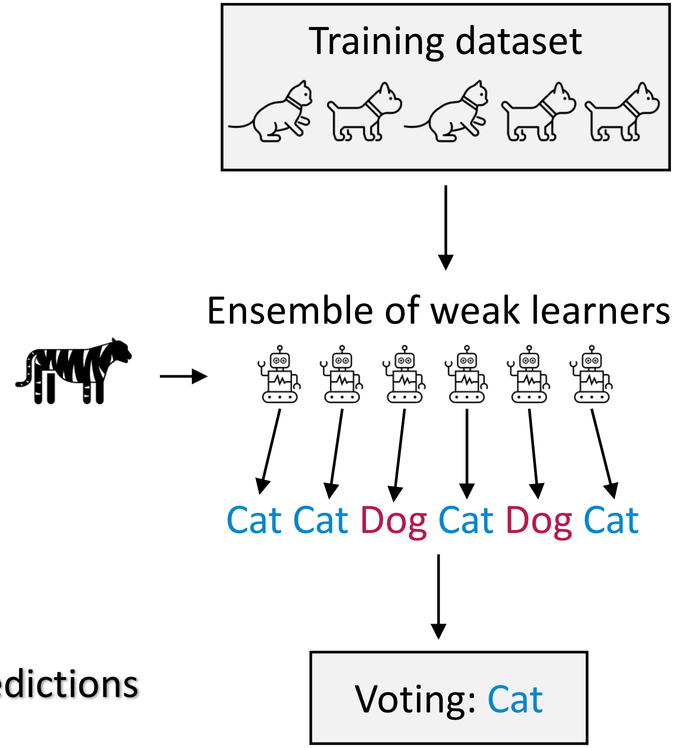


Image source: https://en.wikipedia.org/wiki/Bias–variance_tradeoff

Ensemble Learning

- Why ensemble learning work?
  - A binary classification problem
  - Suppose there are 25 independent classifiers
    - Weak learner, each with an error rate $\epsilon = \mathbf{0.35}$
  - Generate prediction through **major voting**
    - The probability that the ensemble makes a wrong prediction is
      - $P(X \geq 13) = \sum_{i=13}^{25} \binom{25}{i} \epsilon^i (1-\epsilon)^{25-i} \approx \mathbf{0.06}$
        - ϵ^i is the probability of i weak learners make incorrect predictions
        - $(1-\epsilon)^{25-i}$ is the probability of $(25-i)$ weak learners make correct predictions
        - $\binom{25}{i}$ is the number of ways of choosing a subset of size i from 25 weak learners



Ensemble Learning

- Two questions to answer when designing ensemble learning methods
  1. How to generate multiple base models (estimators)?
  2. How to integrate / combine their predictions to make final prediction?
- Different decisions lead to different **types** of ensemble learning methods
  - Bagging
  - Boosting
  - Stacking

Ensemble Learning

- Two questions to answer when designing ensemble learning methods
  1. **How to generate multiple base models (estimators)?**
     - These base models should be **different**.
       - Given the same input, they should make different predictions (errors).
     - **Training data + ML algorithm → ML model**
       - Different training data
         - Different parts of the original training dataset → **Bagging**
         - Different weights for different samples in the training dataset → **Boosting**
       - Different ML algorithm → **Stacking**
  2. How to integrate / combine their predictions to make final prediction?

Ensemble Learning

- Two questions to answer when designing ensemble learning methods
  1. How to generate multiple base models (estimators)?
  2. **How to integrate / combine their predictions to make final prediction?**
     - For bagging / boosting
       - For regression: Use median, average, weighted average, etc. as the final prediction
       - For classification: Make final prediction through majority voting or soft (weighted) voting
     - For stacking
       - Train another ML model to make final prediction
         - Inputs: The prediction of each base model
         - Output: The final prediction

Outline

- Ensemble Learning
- **Bagging**
- Bias & Variance
- Random Forests

Bagging

- **Bootstrap Aggregating → Bagging**
- An ensemble learning technique
  - **Parallel model training**
    - Multiple models / estimators are trained simultaneously on different subsets of the original training dataset.
  - **Bootstrap sampling**
    - Each subset of training data is generated by randomly sampling with replacement.
  - **Aggregation of predictions**
    - Regression: Averaging the predictions
    - Classification: Majority voting

Bagging

- **Bootstrap sampling**
  - A statistical resampling where random samples of the training dataset are taken **with replacement.**

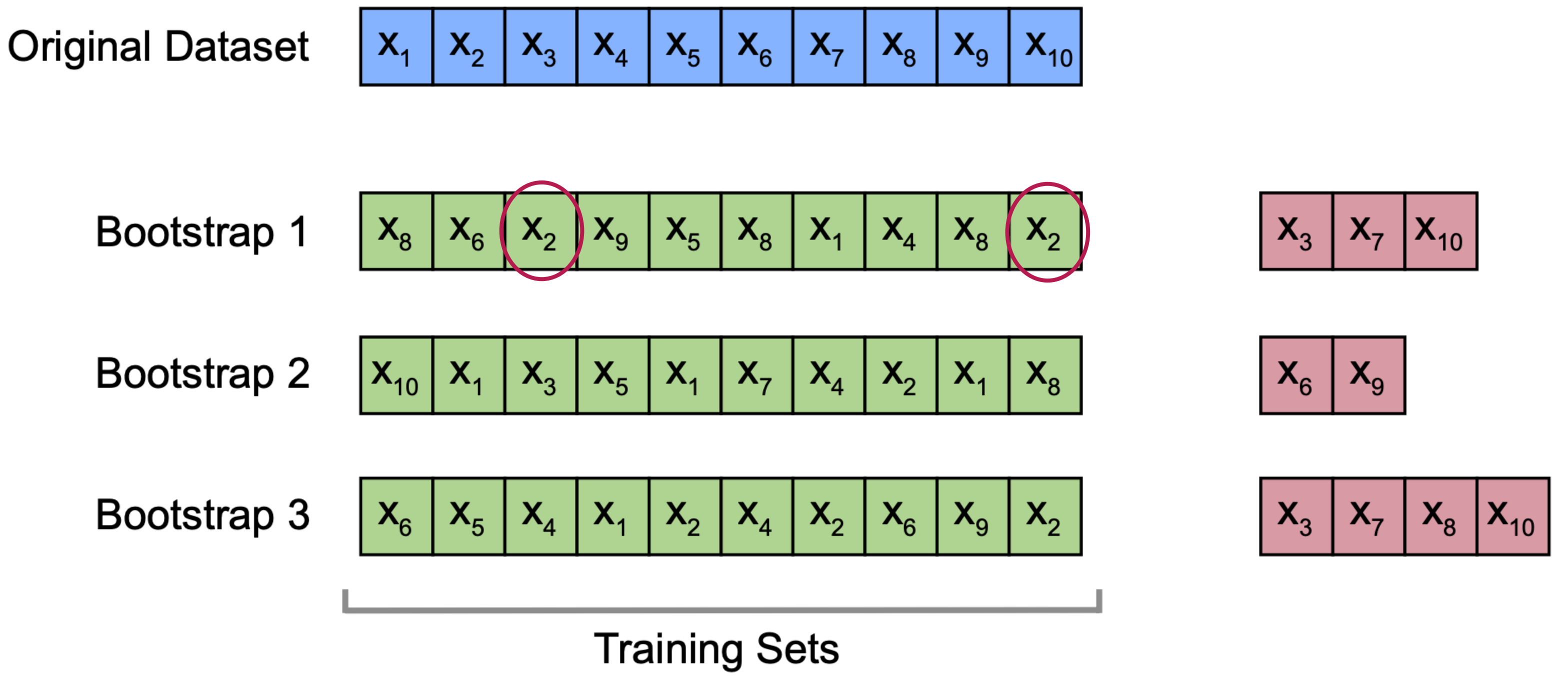
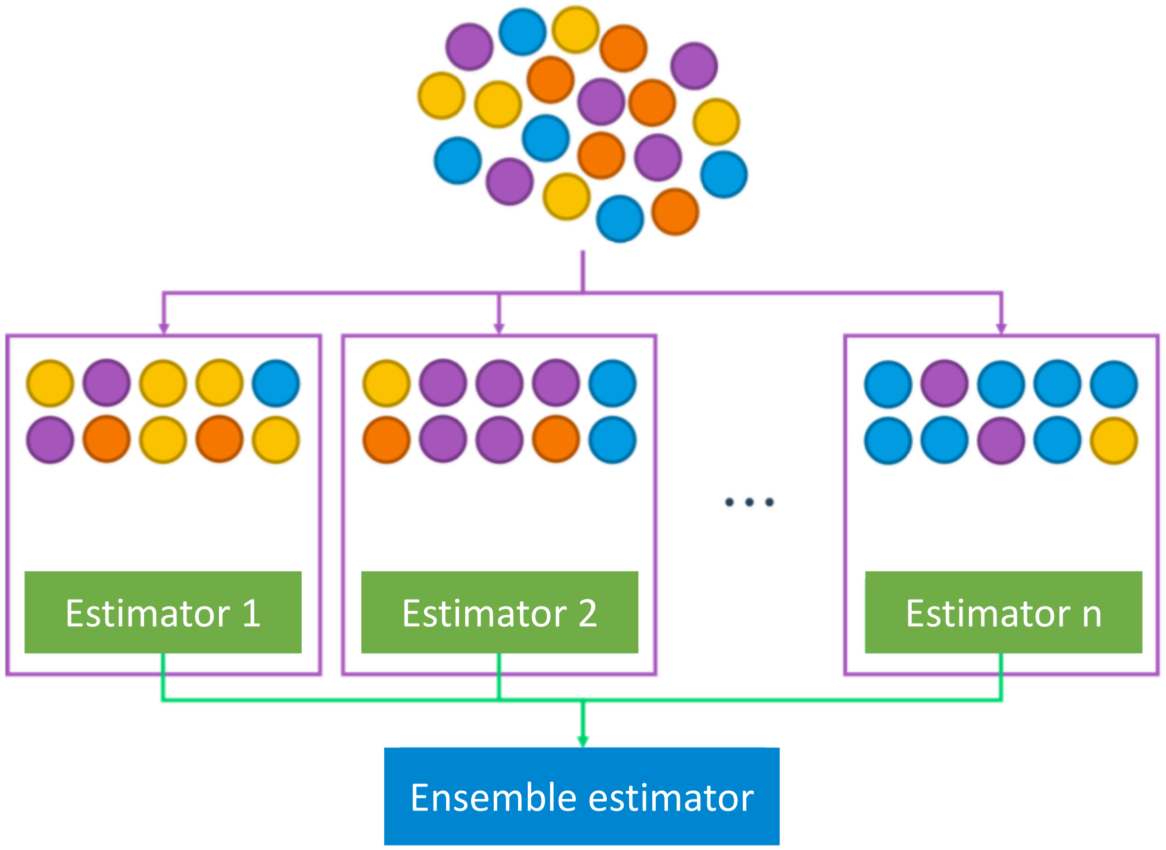


Image source: Slides from Sebastian Raschka @ University of Wisconsin-Madison

Bagging



Original dataset

Bootstrap sampling

Parallel training

Aggregation of predictions

Image source: https://en.wikipedia.org/wiki/Bootstrap_aggregating

Outline

- Ensemble Learning
- Bagging
- **Bias & Variance**
- Random Forests

Bias & Variance

- How bagging reduce the error of single model?
- The bias & variance intuition
  - Suppose we use the same ML algorithm, but different subsets of original dataset to train multiple estimators
  - Task: Predict the bullseye

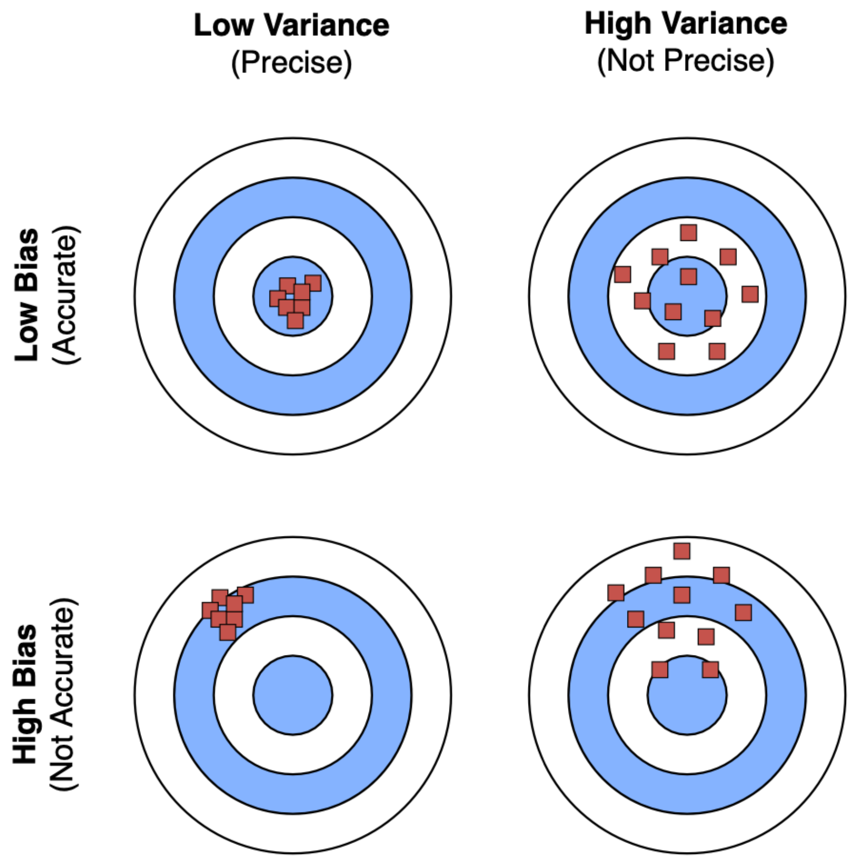
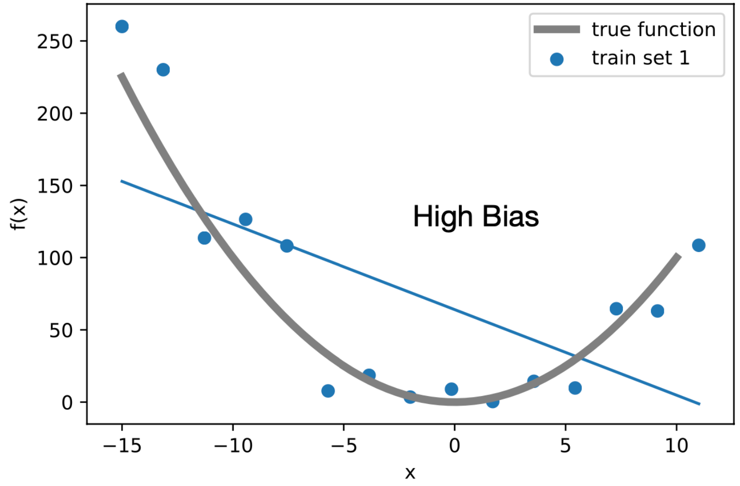


Image source: Slides from Sebastian Raschka @ University of Wisconsin-Madison

Bias & Variance

- **Bias**
  - The error from erroneous (often overly simplistic) assumptions in the algorithm.
  - High bias: The model failed to learn the hidden pattern between inputs and output.
  - **Under-fitting**



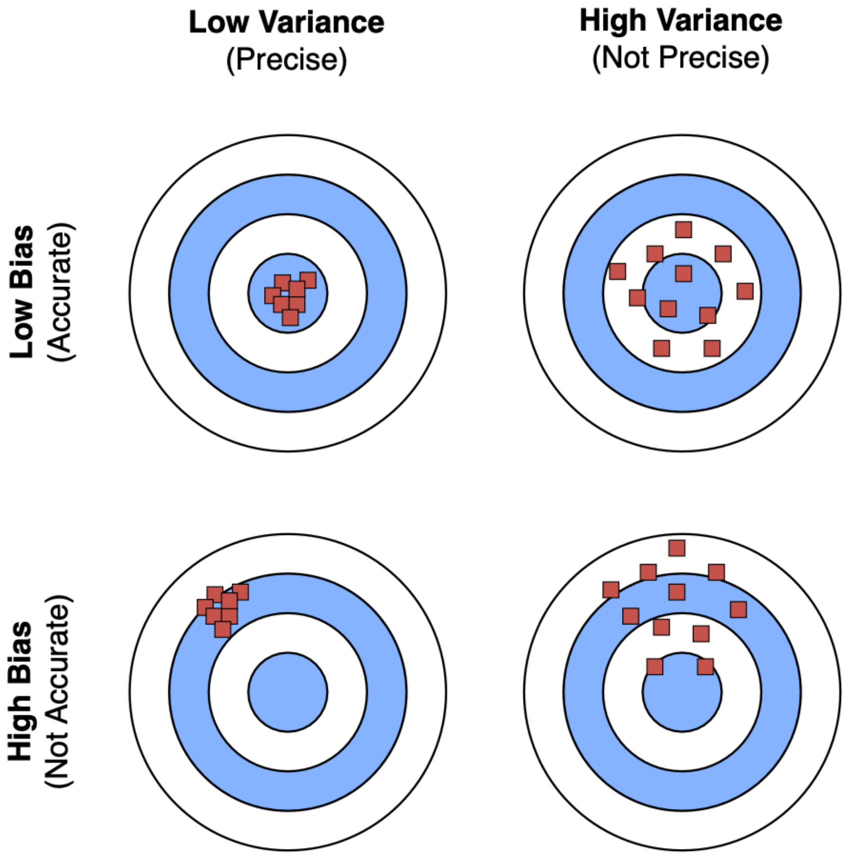
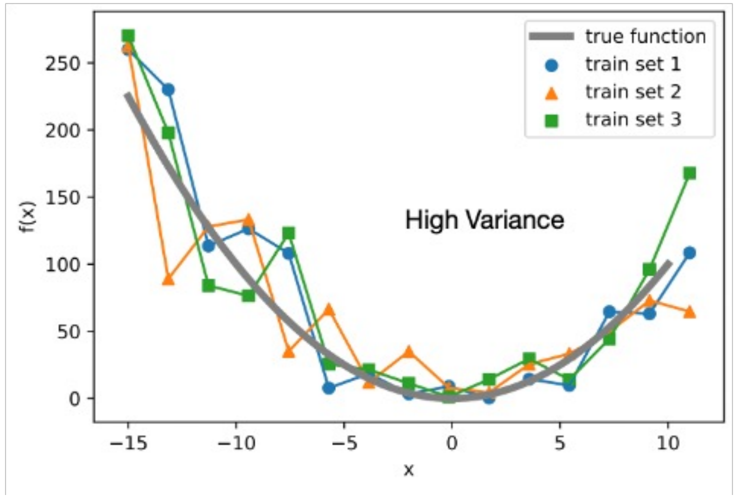


Image source: Slides from Sebastian Raschka @ University of Wisconsin-Madison

Bias & Variance

- **Variance**
  - Variability of predictions from one base estimators to another.
  - High variance: The model is sensitive to the random noise in the training data.
  - **Over-fitting**



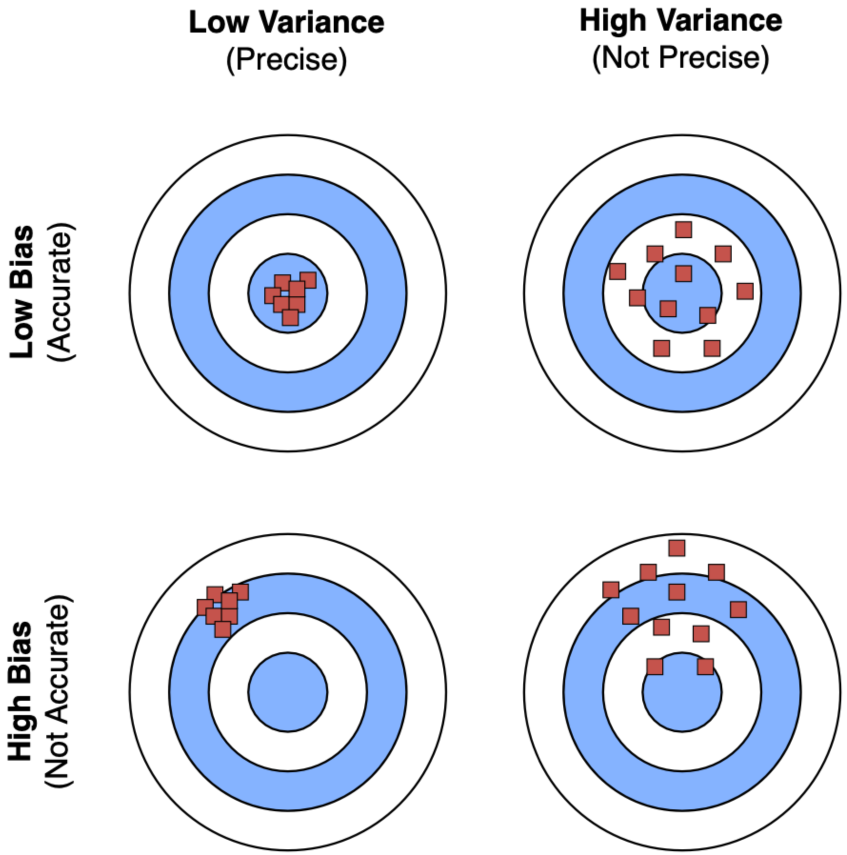


Image source: Slides from Sebastian Raschka @ University of Wisconsin-Madison

Bias & Variance

- Bias-Variance trade-off
  - For single model

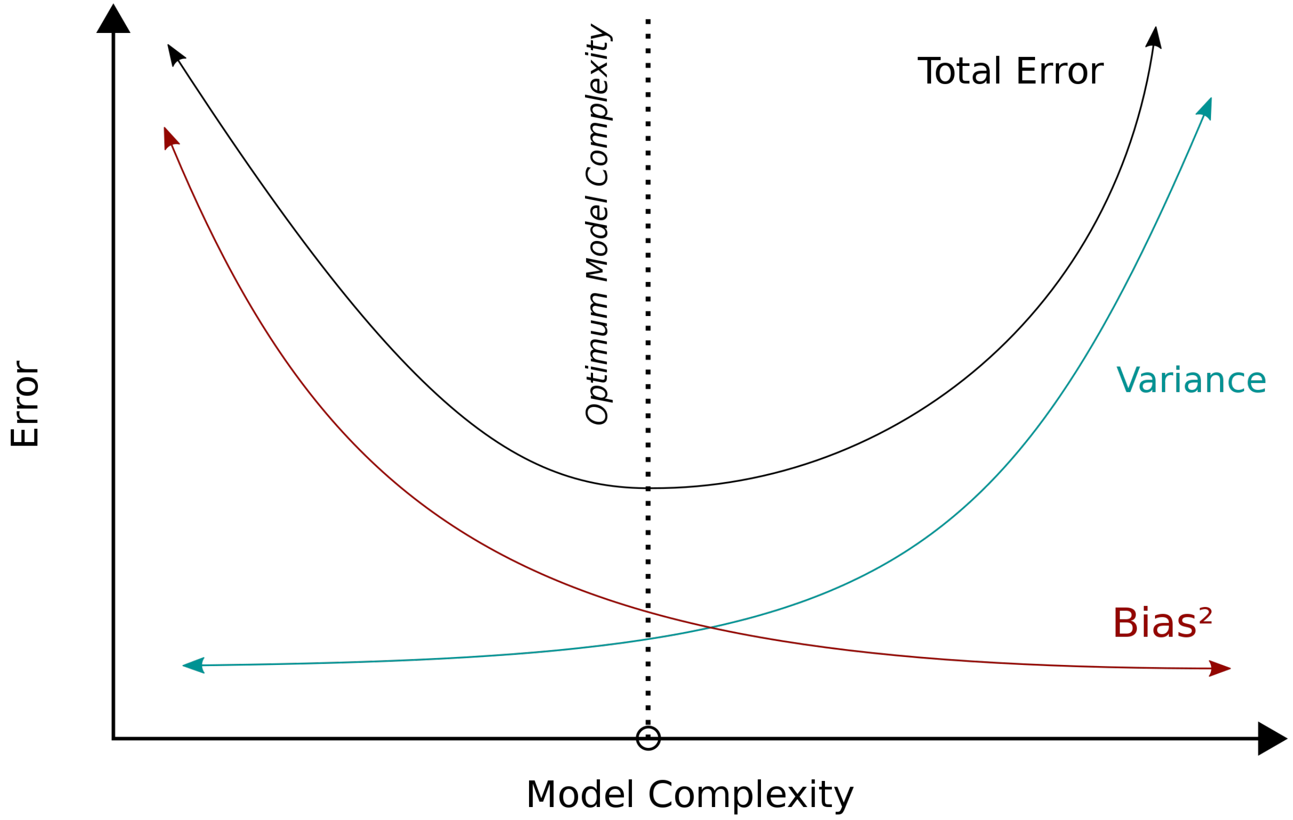


Image source: https://en.wikipedia.org/wiki/Bias–variance_tradeoff

Bias & Variance

- Bias-Variance trade-off
  - For ensemble learning – Bagging
    - Generate multiple base estimators with
      - Low bias
      - High variance
    - Average the predictions to reduce the high variance and keep the low bias

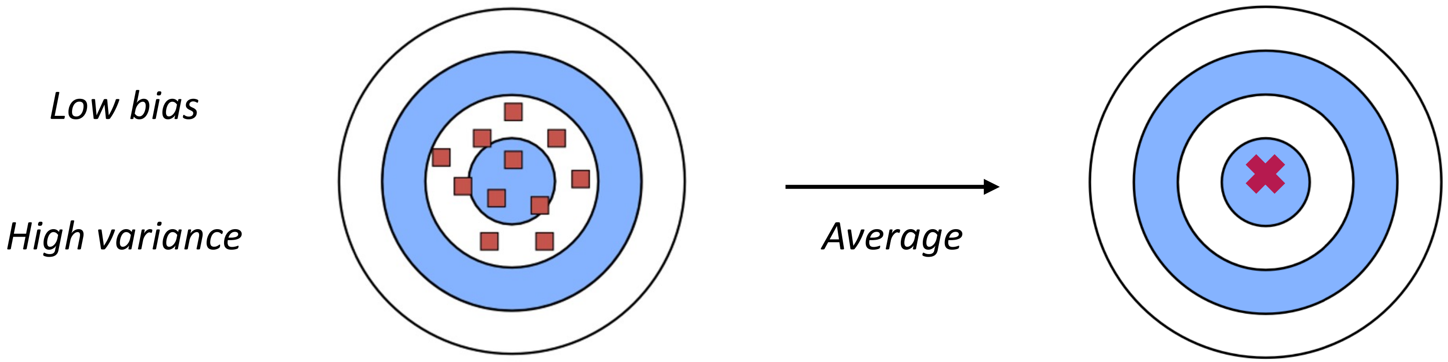


Image source: https://en.wikipedia.org/wiki/Bias–variance_tradeoff

Bias & Variance

- Bias-Variance trade-off
  - For ensemble learning – Bagging
    - Generate multiple base estimators with
      - **Low bias**
      - **High variance**

      → *Increase model complexity* → *Over-fitted models* → *Decision Tree*
    - Average the predictions to reduce the high variance and keep the low bias

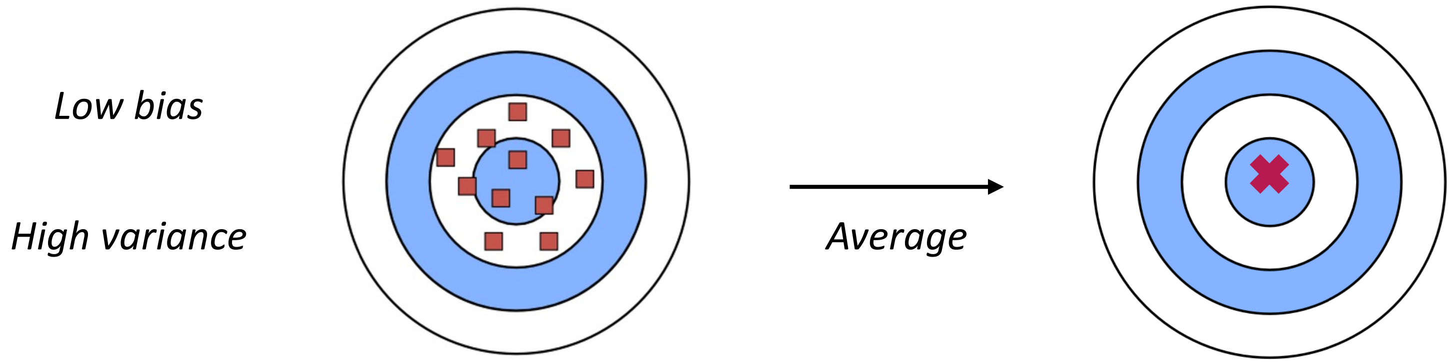


Image source: https://en.wikipedia.org/wiki/Bias–variance_tradeoff

Outline

- Ensemble Learning
- Bagging
- Bias & Variance
- **Random Forests**

Random Forests

- An ensemble learning algorithm
- A bagging method that uses decision tree as the weak learner
- Why decision tree?
  - Prone to over-fitting
  - High variance, low bias
- One more trick
  - When training each decision tree, use a random subset of features instead of all features.
    - Make different base estimators less correlated, more independent

Random Forests

- Implementation in scikit-learn
  - sklearn.ensemble.RandomForestClassifier
  - sklearn.ensemble.RandomForestRegressor
- Key hyper-parameter
  - Key hyper-parameters of decision tree algorithm
    - Criterion
    - Max_depth
    - Min_samples_split
    - Min_samples_leaf
    - Min_impurity_decrease

Random Forests

- Key hyper-parameter

n_estimators : *int, default=100*

The number of trees in the forest.

bootstrap : *bool, default=True*

Whether bootstrap samples are used when building trees. If False, the whole dataset is used to build each tree.

max_samples : *int or float, default=None*

If bootstrap is True, the number of samples to draw from X to train each base estimator.

- If None (default), then draw `X.shape[0]` samples.
- If int, then draw `max_samples` samples.
- If float, then draw `max(round(n_samples * max_samples), 1)` samples. Thus, `max_samples` should be in the interval `(0.0, 1.0]`.

Random Forests

- Byproduct
  - Feature importance
    - Decision tree can calculate the importance of a feature based on how much it contributes to reducing the weighted average impurity (Gini or entropy) in the tree.
    - Random Forests averages the feature importance across all trees in the forest to obtain the final importance score for each feature
    - The raw importance scores can be normalized to sum up to 1.
    - Additional insights from feature importance
      - Identify key features
      - Feature selection
      - Domain insights

Hands-on Exercise

- Exercise 01