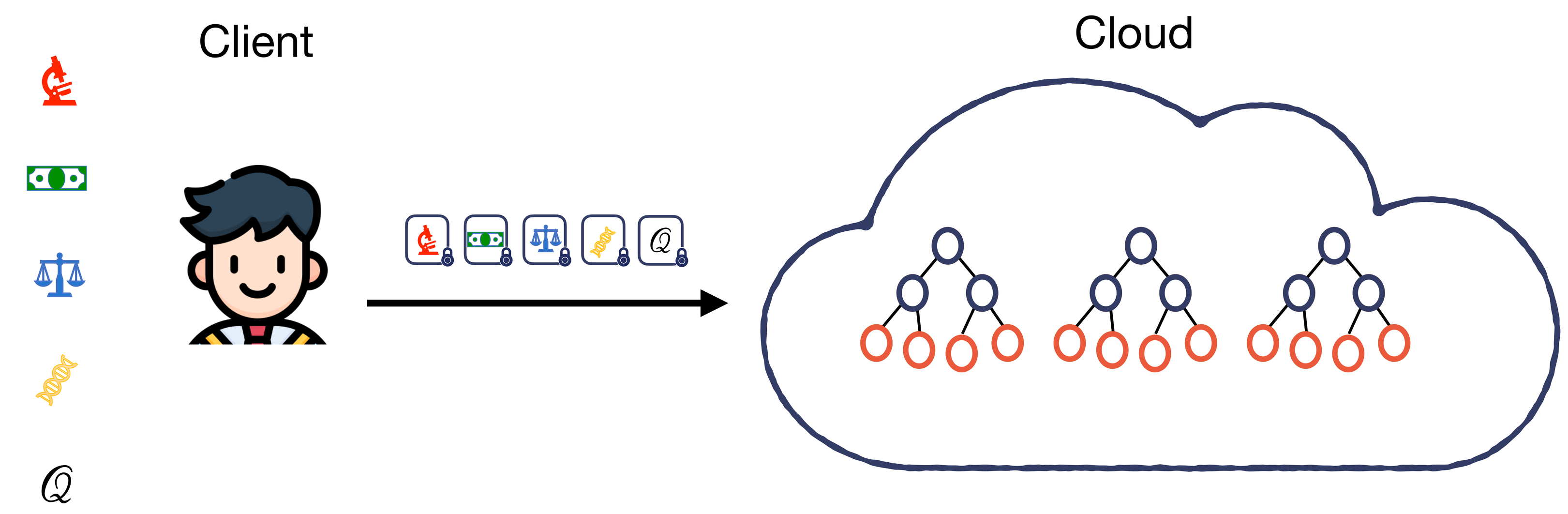
Oblivious (Un)Learning of Extremely Randomized Trees

Sofiane Azogagh, Zelma Aubin Birba, Marc-Olivier Killijian, Sébastien Gambs
azogagh.sofiane@uqam.ca
UQÀM DEEL DEpendable & Explainable Learning

Context



This work explores privacy-preserving Machine Learning as a Service where a Client sends sensitive data to a Cloud provider (e.g AWS, Google Cloud etc..) to train a model. Alongside the data, the Client also sends an encrypted request $\mathcal{Q}$, indicating whether the operation is **Training** or **Unlearning**. Using **Fully Homomorphic Encryption (FHE)**, the Cloud can process the request directly on encrypted data, preserving both the privacy of the data and the nature of the operation.

Extremely Randomized Trees (ERTs)

A **forest** $\mathcal{F}$ of ERTs is defined by several **binary trees** $\mathcal{T}_j$ such as

$$\mathcal{T}_j = \{\mathcal{N}_i\}_{i=1}^{2^{d+1}} \cup \{\mathcal{L}_i\}_{i=1}^{2^d}$$

where $\mathcal{N}_i$ are the internal nodes, $\mathcal{L}_i$ the leaves and d is the depth of $\mathcal{T}_j$.
Each **internal node** $\mathcal{N}_i$ contains a threshold θ_i and a feature index I_i randomly sampled; more formally :

$$\forall i \in \{1,\ldots,2^{d+1}\}, \mathcal{N}_i = (\theta_i, I_i) \xleftarrow{\$} \mathbb{Z}_N^2$$

Each **leaf** $\mathcal{L}_i$ stores the class counts c_k of the training samples that reached it; more formally :

$$\forall i \in \{1,\ldots,2^d\}, \mathcal{L}_i = (\,|\,c_0\,|\,,\ldots,\,|\,c_\ell - 1\,|\,)$$

where ℓ is the number of possible classes.

Oblivious Operations

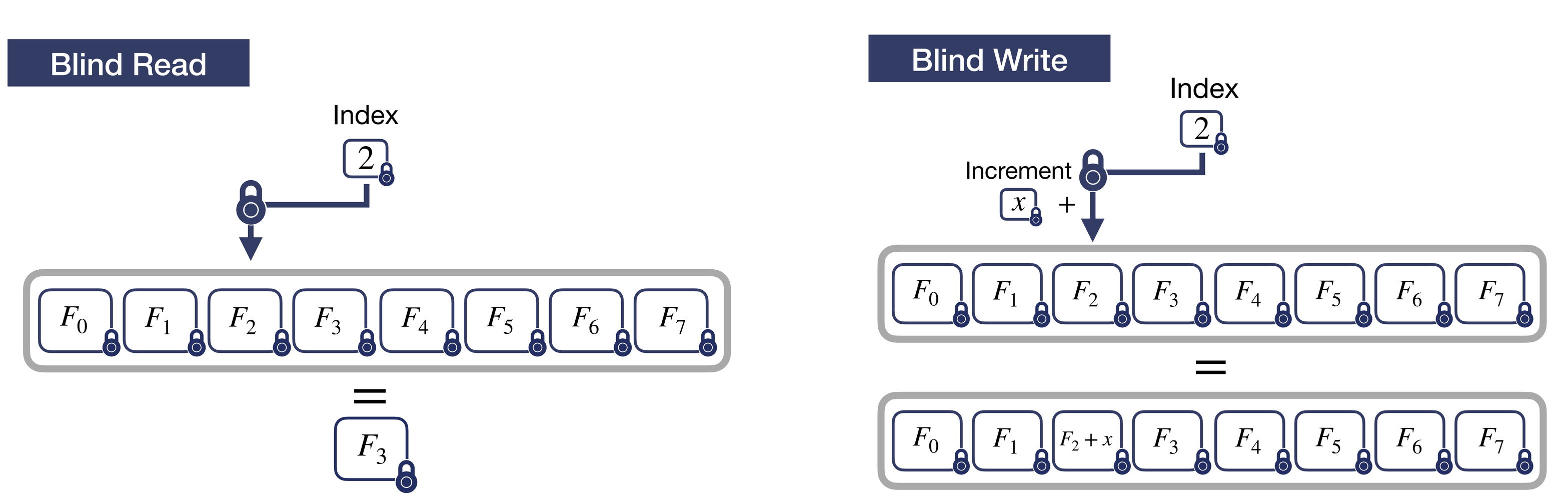
The *tfhe-rs*[1] library implements the TFHE scheme which allows to manipulate encryptions of integers from $\mathbb{Z}_p$. We built **RevoLUT[2]** on it to support oblivious manipulation of encrypted arrays of up to p elements.

Basic operations

Addition: $x + y = x+y$

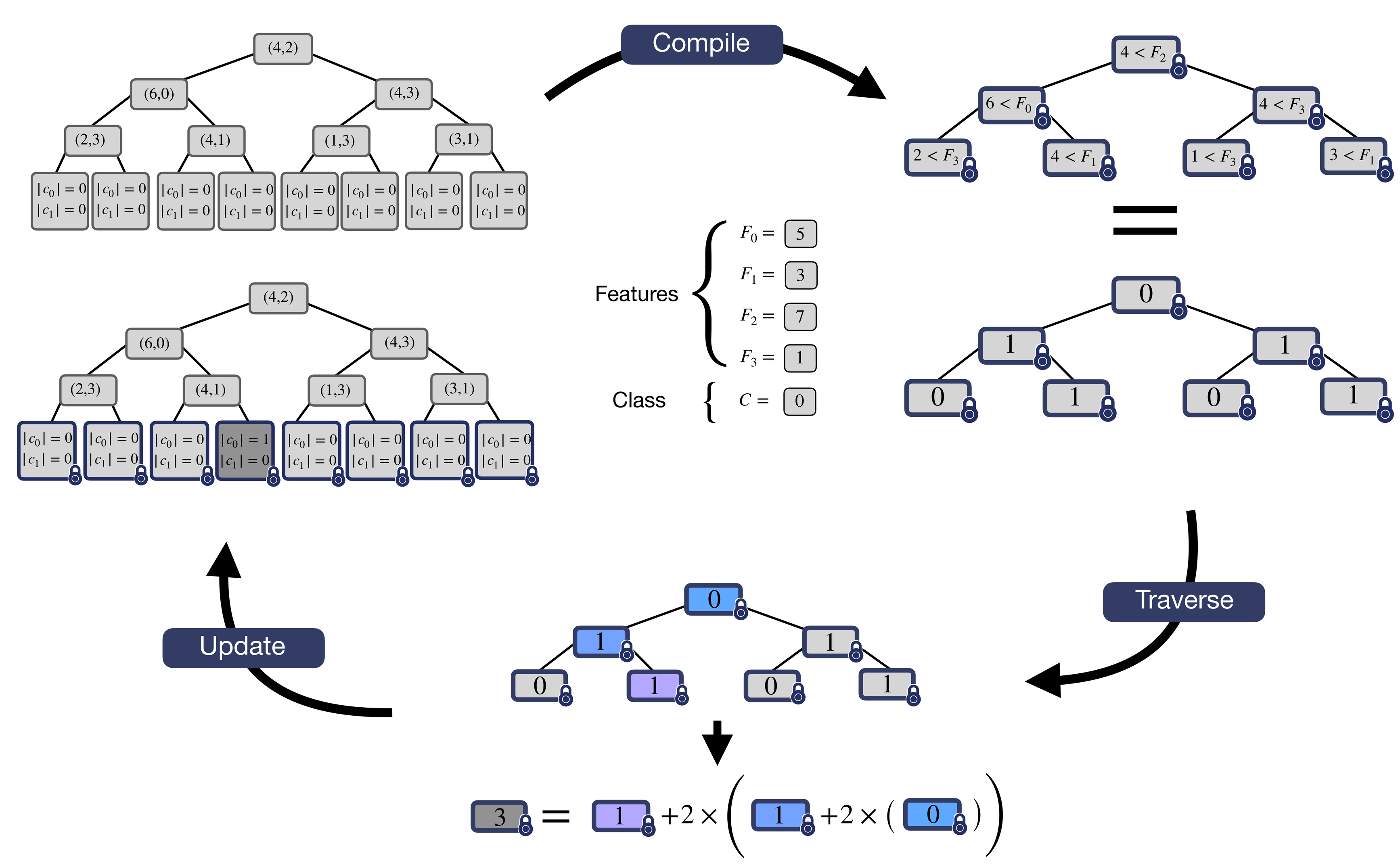
Scalar Multiplication: $x \times y = x \times y$

RevoLUT operations



Our protocol

The server generates a forest with a predefined number of trees. Each tree is a binary tree of a predefined common depth d, with features and samples randomly selected among all the features and their potential values in the dataset.



For each sample $S = (F_0, \ldots, F_n, C)$, the following operations are executed

1. Compile

At each node $\mathcal{N}_i$, the corresponding sample feature F_{I_i} is **blindly** compared to the node's feature threshold θ_i to produce an **encrypted comparison result**.

2. Traverse

At each level of the tree, the **encrypted comparison result** of the reached node is used to select the left or right child in the following level until a leaf is reached.

3. Update

In the **blindly reached leaf**, we **blindly** update the counter corresponding to the label C of the sample. The counter gets **incremented or decremented**, based on whether the request $\mathcal{Q}$ is for learning or unlearning. If a counter gets larger than p, an **overflow** phenomenon occurs (see Fig. 2).

Experimental results

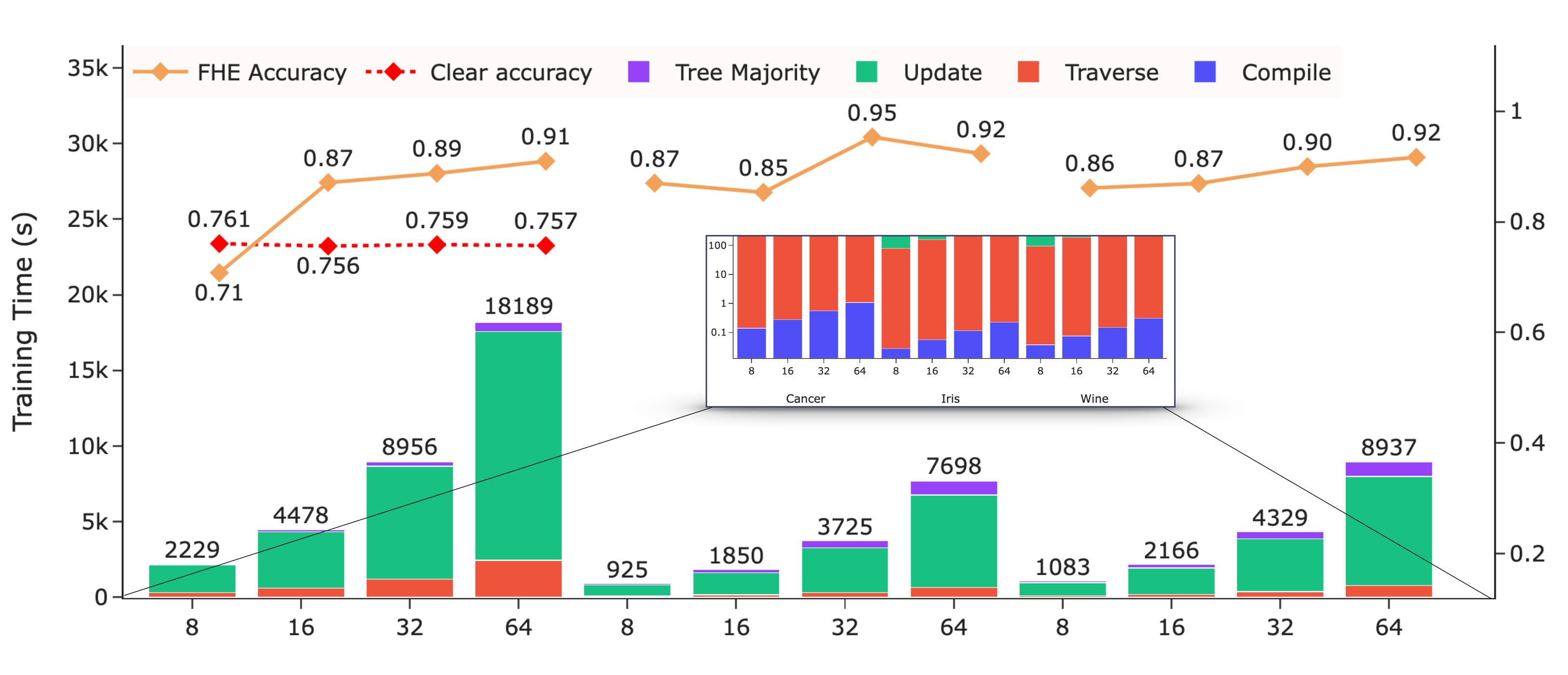


Fig. 1 : Training time and accuracies across various forest sizes $\mathcal{F}$ (i.e 8, 16, 32 and 64) and different datasets.

Fig. 2 : Accuracy of different forest $\mathcal{F}$ during training on two datasets and their respective PCA projections to illustrate the class separability.

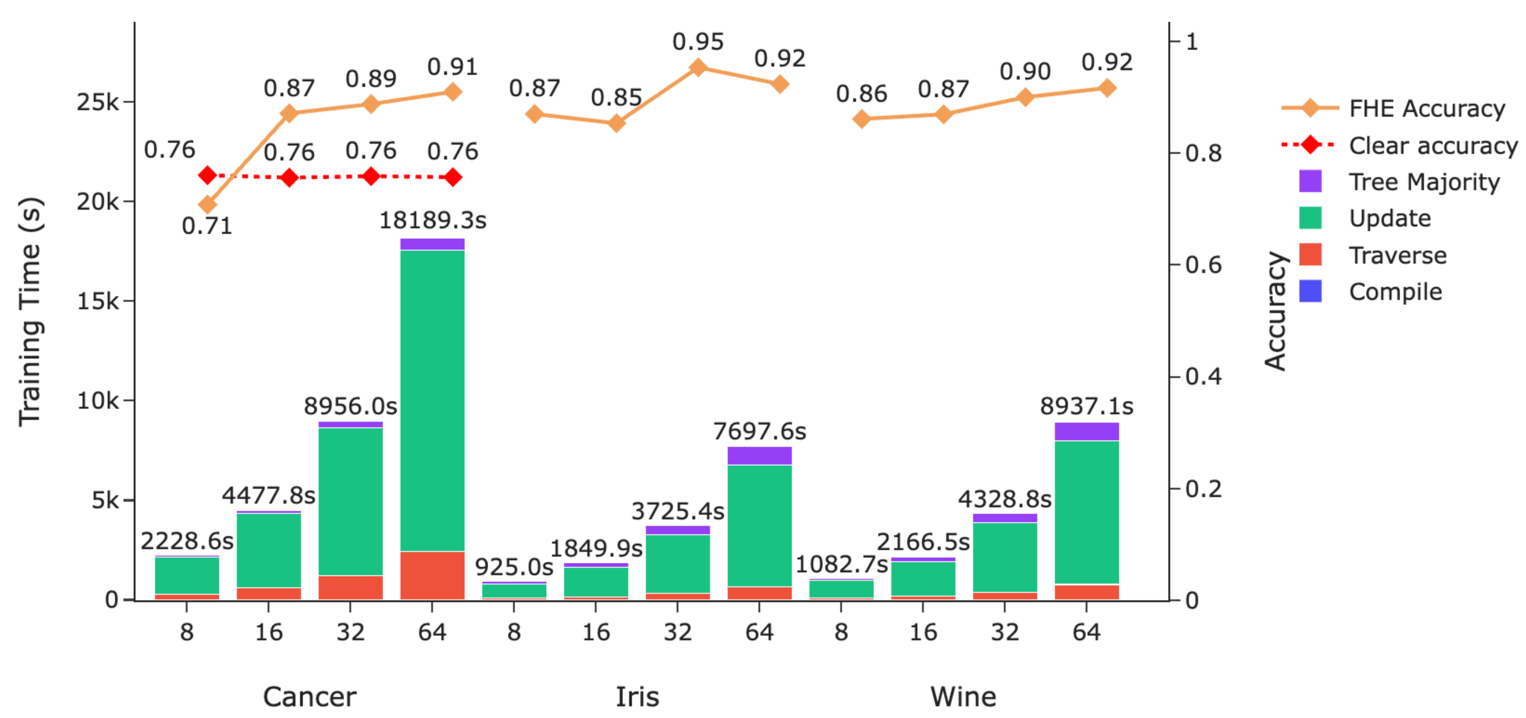
Key takeaways

- We provide the first homomorphic learning algorithm on ERTs, which also supports oblivious unlearning.
- ERTs provide competitive accuracy and even improve it under certain conditions that favor the occurence of counter overflow.

References

[1] Zama. TFHE-rs: A Pure Rust Implementation of the TFHE Scheme for Boolean and Integer Arithmetics Over Encrypted Data. 2022

[2] S. Azogagh, Z. A. Birba, M.-O. Killijian, F. Larose-Gervais, and S. Gambs. RevoLUT : Rust efficient versatile oblivious look-up-tables. 2025



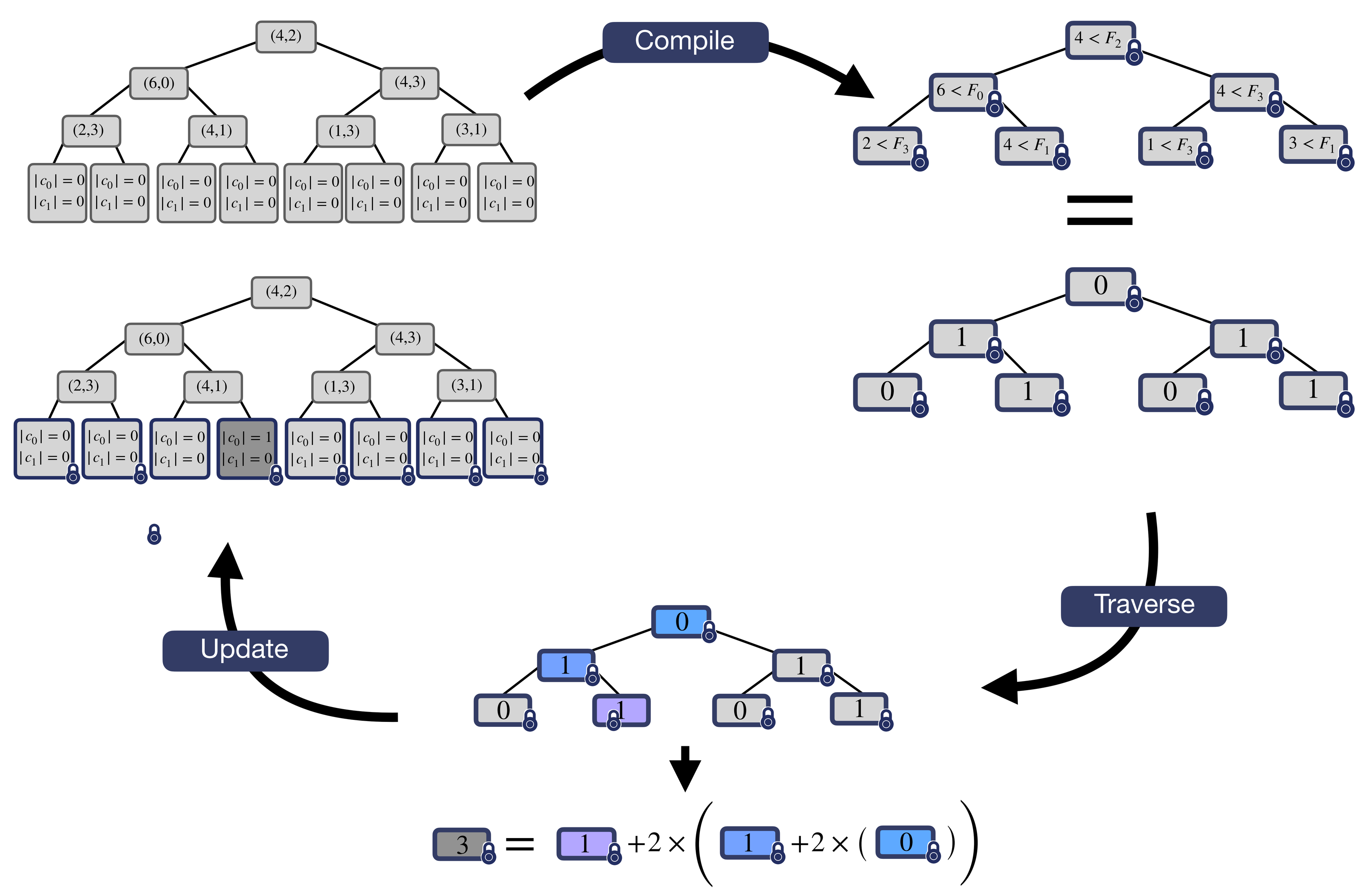


Fig. 2 : Accuracy of different forest $\mathscr{F}$ during training on two datasets and their respective PCA projections to illustrate the class separability.

(a) Japanese Credit Score

(b) Breast Cancer