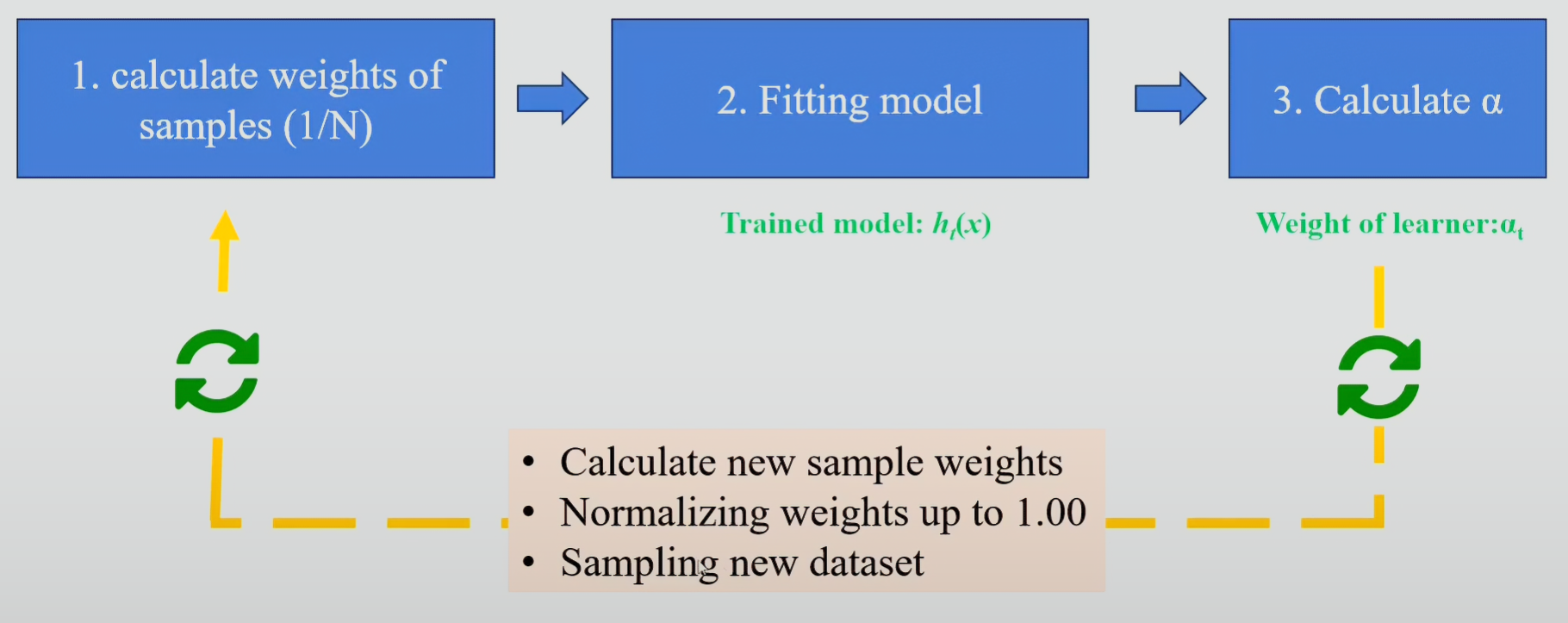
**HW: Adaboost**

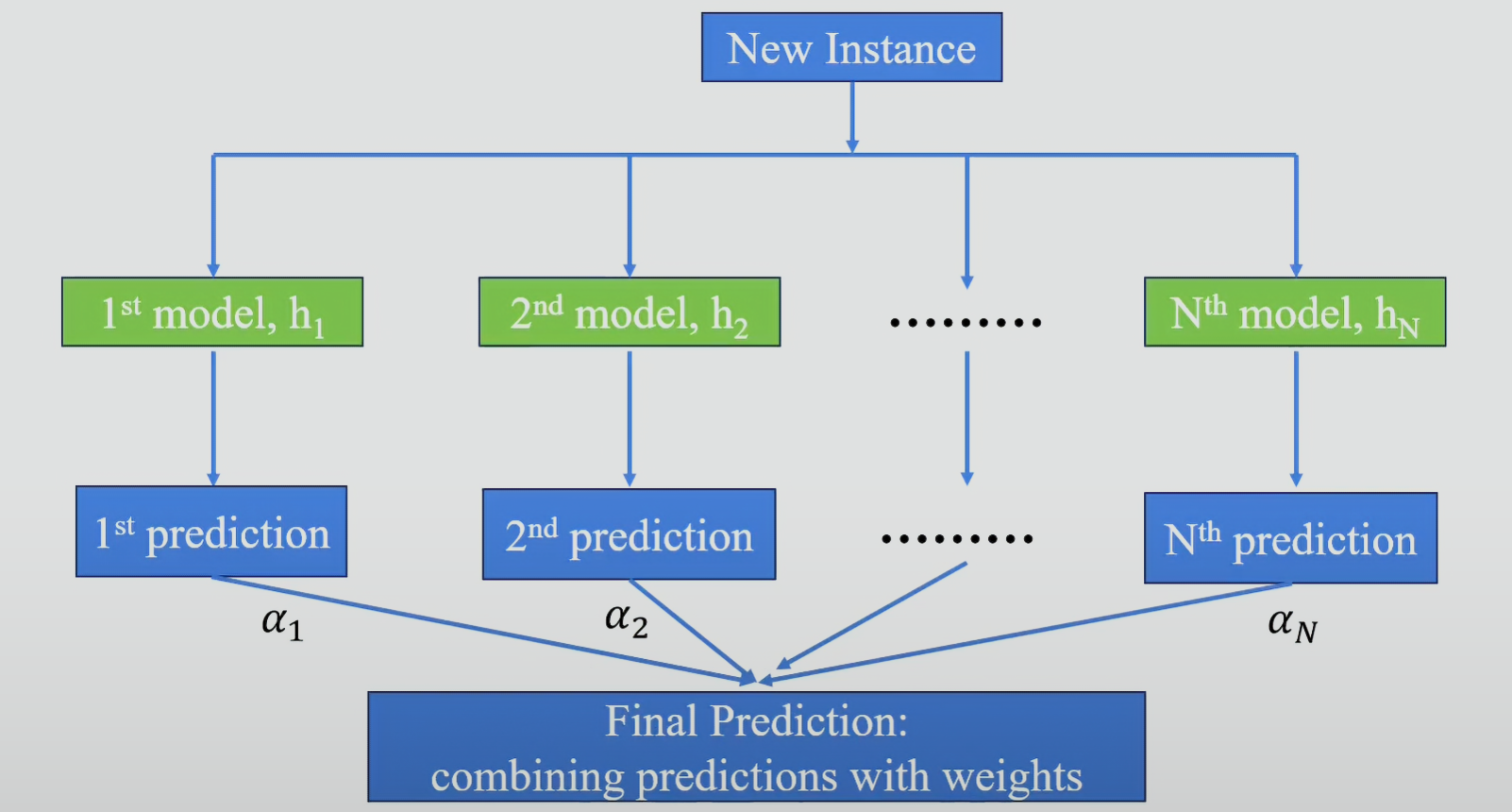
1. Đọc và hiểu overview về thuật toán boosting AdaBoost đã học trên lớp.

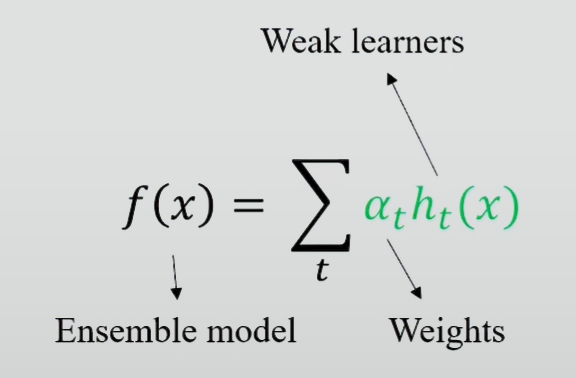
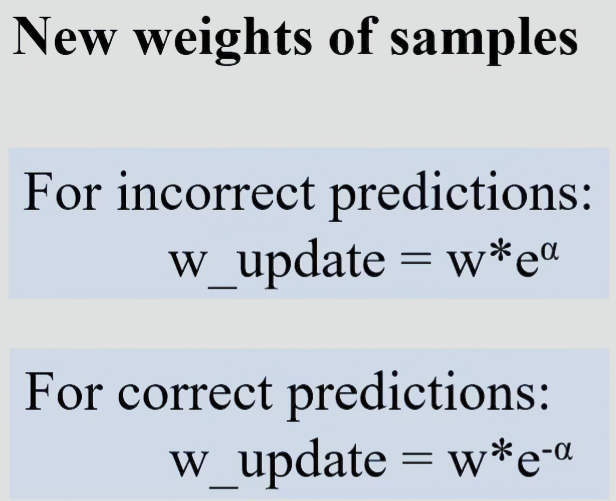
* Cho Flow Chart cách thức training, inference của AdaBoost cùng các thành phần khác như hình dưới:

Training



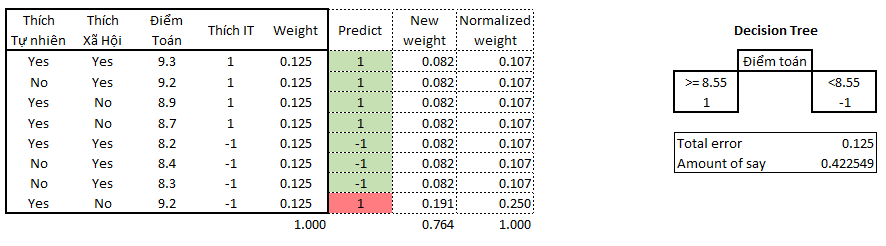
Inference



* Thông tin bổ sung:
* Tại bước 2 của Flow chart: fit model bằng decision tree như đã học
* Tại bước 3 của Flow chart: tính alpha chính là tính “amount of say” như đã học bằng công thức:
* Công thức tính “New weights of samples” được sử dụng để tính lại weights mới cho toàn bộ các điểm trong bộ dữ liệu trước đó (1/N). Lưu ý khi tính lại weight cho các sample, tổng weights của tất cả các sample sẽ != 1 nên cần normalize lại tổng về 1.
* Hàm số “Ensemble model” tượng trưng cho việc sau khi tính toán ra đủ số lượng “Weak learners”, ta tiến hành sử dụng toàn bộ những dự đoán của các “Weak learners” để đưa ra kết quả cuối cùng.

1. Áp dụng kiến thức đã học về AdaBoost, thực hiện xây Decision Tree gốc và thực hiện Boosting 1 lần cho dataset sau:



* Assign equal weight at first:
* Build stump tree with splitting point of 8.55 Điểm Toán
* Predict, calculate ‘Total error’ and ‘Amount of Say’
* Update weights and normalize new weights.

1. LightGBM nhanh và vẫn đạt độ chính xác cao do model được sử dụng các kỹ thuật sau:

* Histogram-Based Learning (binning dataset)

Instead of finding the split points on the sorted feature values, histogram-based algorithm buckets continuous feature values into discrete bins and uses these bins to construct feature histograms during training

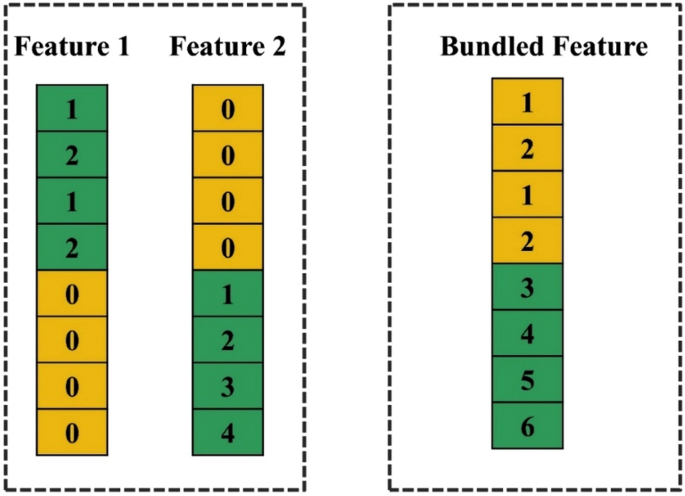
→ efficient in both memory consumption and training speed. It costs ***O(row \* feature)*** for split point finding and ***O(bins \*feature*** ) for histogram building

* Exclusive Feature Bundling

In high dim data, there are a large number of features that are mutually exclusive. Which means that these features rarely take nonzero values simultaneously. Thus we can safely bundle such exclusive features.

For example, after onehotencoder 3 distinct categorical of 1 feature into 3 column, we can bundle 3 columns in to 1 as original.

Or look at 8 observes below, we see that they rarely take nonzero values simultaneously in Feature 1 and 2 → bundle to 1 feature

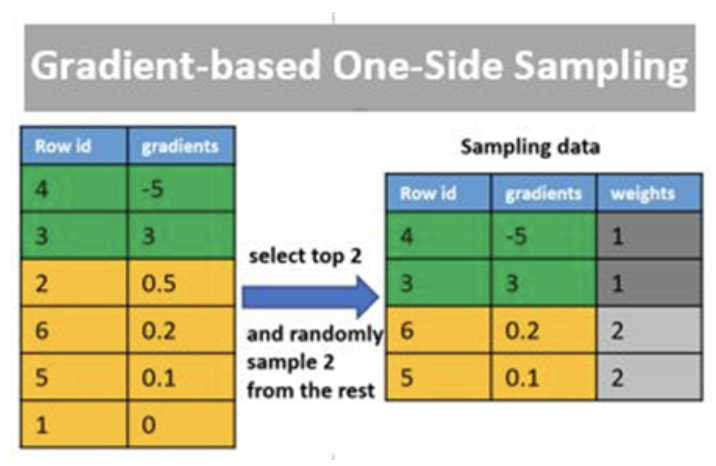


* Gradient-Based One-side Sampling

After each predictions, the good predictions (low error) are well-understood by the algorithm → no need to be retrained, time consuming:

* Remove all → negatively affect on data distribution, ↓ Accuracy
* Down samling is better → faster

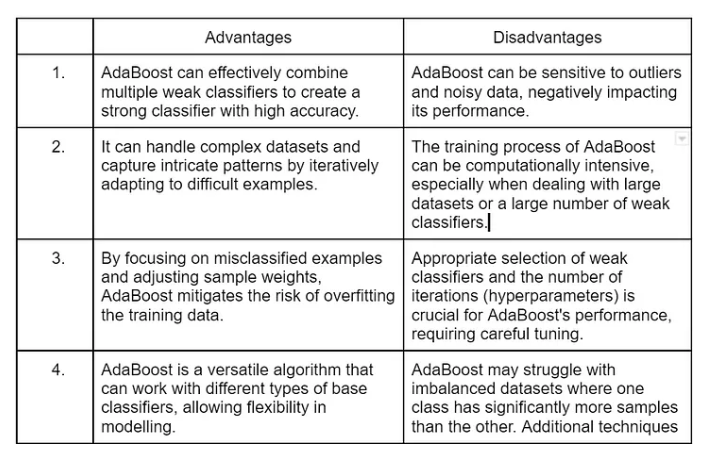
The bad predictions (high error) are remained and weighted higher to be focused by algorthm.



<https://medium.com/@pritmanvar/lightgbm-essentials-how-it-works-and-why-its-fast-586b83dda7af>

1. [Optional] So sánh CatBoost, XGBoost, LightGBM trong 3 aspects sau: Cách decision tree được build; categorical feature được handle như thế nào; Cách sampling

* Với mỗi đặc điểm của mỗi thuật toán phải hiểu sơ lược và giải thích được



Step 1: When the algorithm is given data, it starts by Assigning equal weights to all training examples in the dataset. These weights represent the importance of each sample during the training process.

Step 2: Here, this algorithm iterates with a few algorithms for a specified number of iterations (or until a stopping criterion is met). The algorithm trains a weak classifier on the training data. Here the weak classifier can be considered a model that performs slightly better than random guessing, such as a decision stump (a one-level decision tree).

Step 3: During each iteration, the algorithm trains the weak classifier on given training data with the current sample weights. The weak classifier aims to minimize the classification error, weighted by the sample weights.

Step 4: After training the weak classifier, the algorithm calculates classifier weight based on the errors of the weak classifier. A weak classifier with a lower error receives a higher weight.

Step 4: Once the calculation of weight completes, the algorithm updates sample weights, and the algorithm gives assigns higher weights to misclassified examples so that more importance in subsequent iterations can be given.

Step 5: After updating the sample weights, they are normalized so that they sum up to 1 and Combine the predictions of all weak classifiers using a weighted majority vote. The weights of the weak classifiers are considered when making the final prediction.

Step 6: Finally, Steps 2–5 are repeated for the specified number of iterations (or until the stopping criterion is met), with the sample weights updated at each iteration. The final prediction is obtained by aggregating the predictions of all weak classifiers based on their weights.