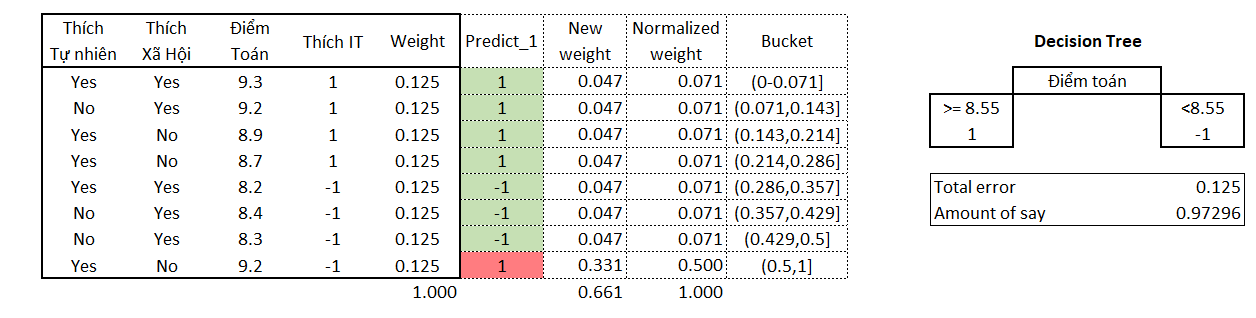
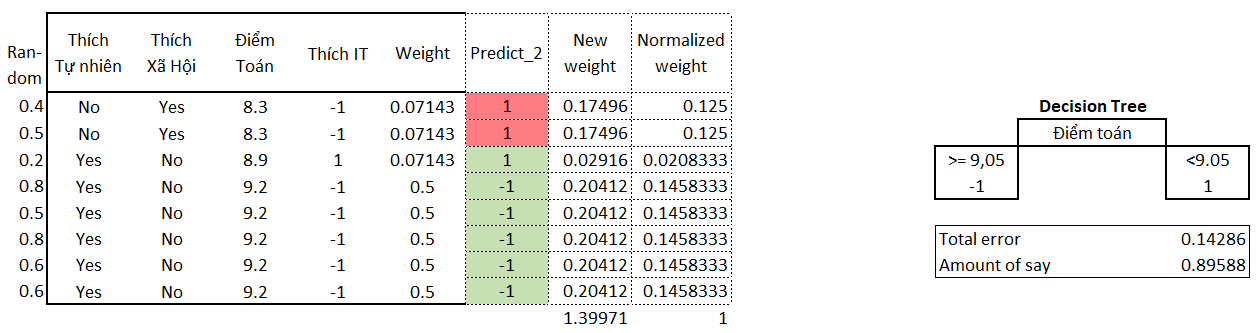
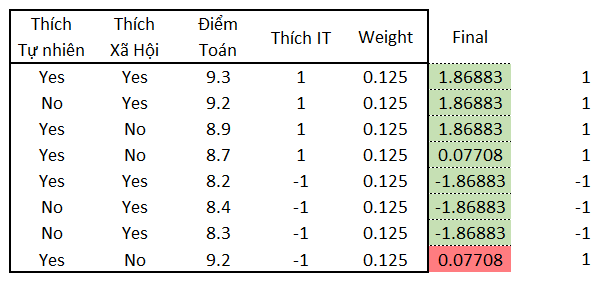
1. Áp dụng kiến thức đã học về AdaBoost, thực hiện xây Decision Tree gốc và thực hiện Boosting 1 lần cho dataset sau:





* 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.

Final prediction:

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

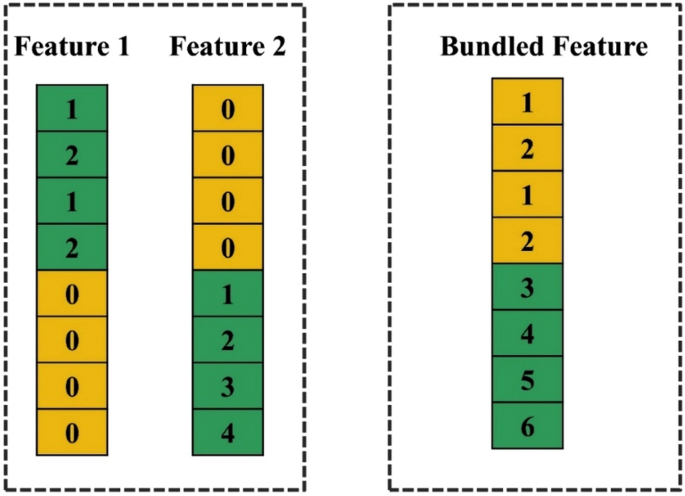
* **Histogram-Based Learning**

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

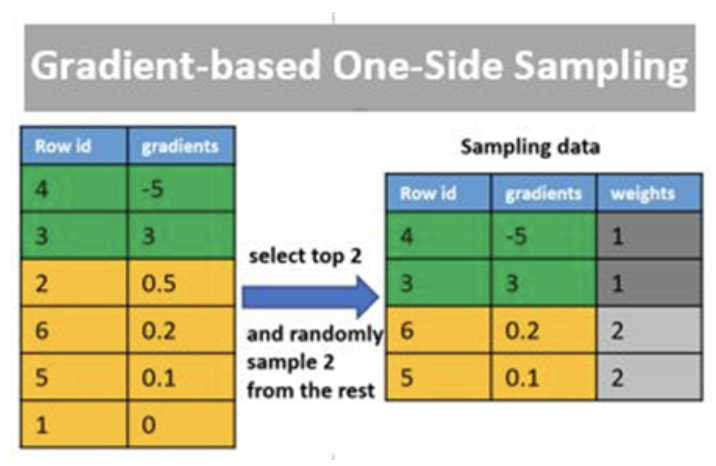
→ efficient in both memory consumption and training speed.

* **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 good predictions will negatively affect on data distribution, and reduce Accuracy. Down samling is better and more suitable.

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

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

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CatBoost** | **XGBoost** | **LightGBM** |
| **Decision Tree** | Grow tree by level, the **same splitting criterion** is used across an entire level of the tree. The feature-split are chosen by its low loss. | Grow tree by depth, then starts pruning the tree backwards and removes splits beyond which there is no positive gain.  Also support leaf-wise tree grow as LightGBM | Grow tree by leave. It chooses to grow the leaf that minimizes the loss, allowing a growth of an imbalanced tree. |
| **Handle categorical data** | In trainning phase  One-hot encoding for features with few categories. Advanced mean encoding (target encoding) for the rest | Not support, user have to preprocess categorical data. | Support ordinal category  Find the best split point in k categories of feature, which then partion features into 2 subsets. |
| **Samling** | No | No | Down sampling with well-learnt data |

[LightGBM](https://medium.com/@pritmanvar/lightgbm-essentials-how-it-works-and-why-its-fast-586b83dda7af)

[Catboost 1,](https://hanishrohit.medium.com/whats-so-special-about-catboost-335d64d754ae) [2,](https://python.plainenglish.io/smart-aspects-of-catboost-algorithm-2720a6de4da6)

[Compare](https://towardsdatascience.com/catboost-vs-lightgbm-vs-xgboost-c80f40662924) 1, [2,](https://medium.com/riskified-technology/xgboost-lightgbm-or-catboost-which-boosting-algorithm-should-i-use-e7fda7bb36bc) [3,](https://medium.com/octave-john-keells-group/xgboost-light-gbm-and-catboost-a-comparison-of-decision-tree-algorithms-and-applications-to-a-f1d2d376d89c)