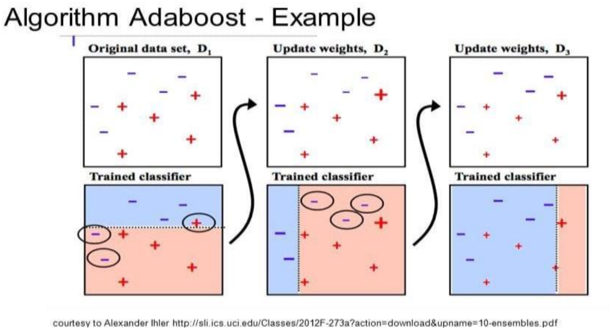
National University of Singapore
School of Computing
Tutorial 6:
Adaptive Boosting and Gradient Boosting

Adaptive Boosting

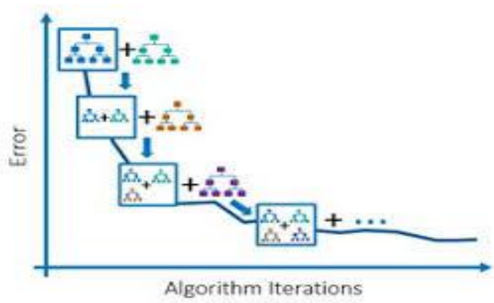
- The weak learners in AdaBoost are decision trees (or any other base classifier that you choose) with a single split, called decision stumps.
- AdaBoost works by putting more weight on difficult to classify instances and less on those already handled well.
- AdaBoost algorithms can be used for both classification and regression problem.



Gradient Boosting

Gradient boosting involves three elements:

- A loss function to be optimized.
- A weak learner to make predictions.
- An additive model to add weak learners to minimize the loss function.



Application: Predict age of abalone using Decision Tree, Adaptive Boosting (AdaBoost) and Gradient Boosting (XGBoost and LightGBM)

1. The UCI Abalone dataset is available from (https://archive.ics.uci.edu/ml/datasets/Abalone). It has been pre-downloaded and made available for this tutorial. The data file is "abalone.data". It can be read into your Jupyter notebook using pandas' read_csv function. The "abalone.names" file contains more information about the dataset, and the names for your headers can be found in this file.

2. Read the dataset into your notebook and manually populate the headers with header names.

3. Pre-process and one hot encode the 'sex' variable, since this variable is categorical.

4. Our target is the 'rings' variable. As there are many values in this column, bin the values into 3 separate bins and label them ('young', 'medium' and 'old'). The head() of the dataframe is given here:

	sex	length	diameter	height	whole_weight	shucked_weight	viscera_weight	shell_weight	rings	bins
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15	middle
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7	young
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9	young
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10	young
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7	young

5. Set up the X and y variables and split your data into the training set and testing set.

6. Fit and predict the y variables using a **standard Decision Tree Classifier**.

7. Fit and predict the y variables using an **ADABoost Classifier** with n_estimators=10, learning_rate=1 and a decision tree base estimator with max_depth=3.

8. Fit and predict the y variables using an **XGBoost Classifier** with max_depth=3, learning_rate=0.1, and n_estimators=100.

9. Fit and predict the y variables using an **LightGBM Classifier** with max_depth=3, learning_rate=0.1, and n_estimators=100.

10. Print the accuracy score for each of these classifiers.

11. Perform a grid search on ADABoost, XGBoost and LightGBM using the following parameters
    'n_estimators': [100, 500, 1000],
    'learning_rate': [0.01, 0.1, 1],
    'max_depth': [1,2,3]

12. Print the accuracy scores on the test set for the best estimators for each classifier from the grid search.