Ensemble Learning

Computational Data Science,
Addis Ababa University

www.aau.edu.et

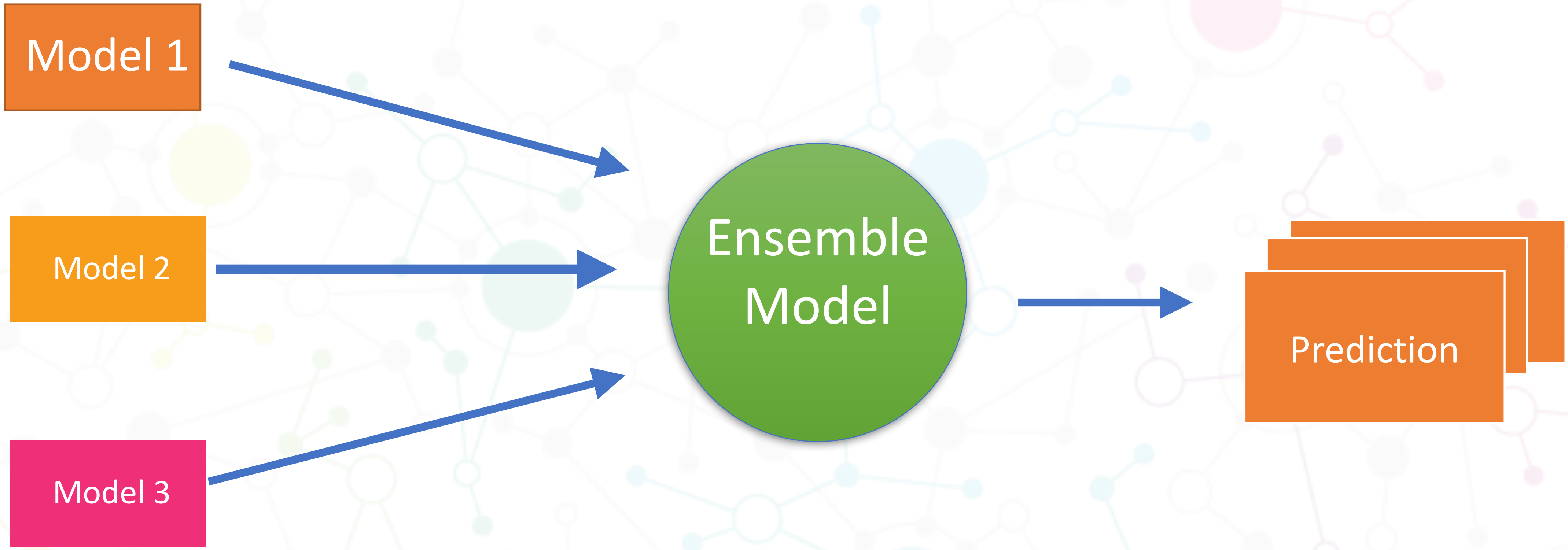
mesfin.diro@aau.eud.et

+251-912-086156

Ensemble Learning

- An *ensemble* is a set of elements that collectively contribute to a whole.
- An ensemble learning techniques combine individual models together to improve the stability and predictive power of the model
- This technique permits higher predictive performance
- It combines multiple machine learning models into one predictive model
- Certain models do well in modeling one aspect of the data, while others do well in modeling another
- Learn several simple models and combine their output to produce the final decision
- The combined strength of the models offsets individual model variances and biases
- This provides a composite prediction where the final accuracy is better than the accuracy of the individual models

Ensemble Learning



- **Ensemble model is the application of multiple models to obtain better performance than from a single model**

Ensemble Learning Categories

Ensemble methods

Sequential methods

Parallel methods

- In sequential ensemble methods the base learners are generated consecutively to use the dependence between the base learners to boot the overall perforce of a model
- Parallel ensemble methods are applied whenever the base learners are generated in parallel to use independence between the base learners. Ex Random forest.

Performance of Ensemble Models

01

Robustness

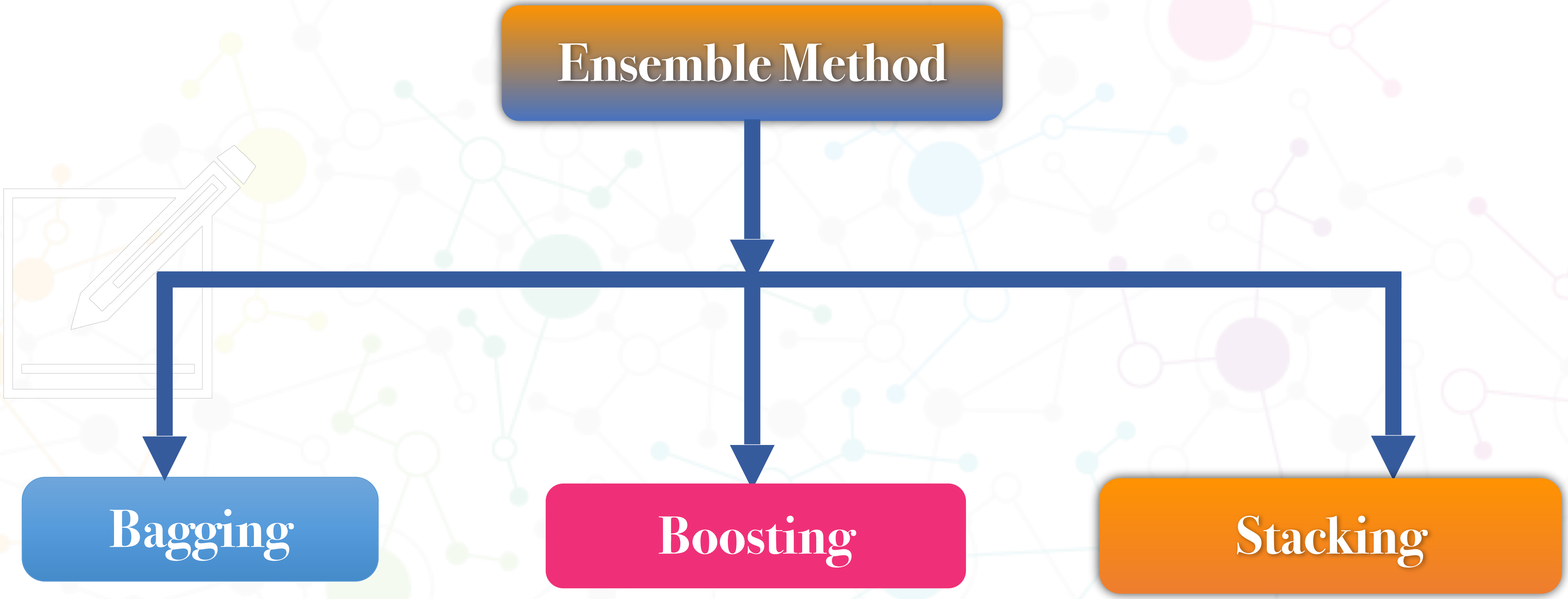
Ensemble models incorporate the predictions from al the base learners

02

Accuracy

Ensemble models deliver accurate predictions and have improved performance

Ensemble Methods



Ensemble Methods

Combine all “weak” learners to form an ensemble or

Create an ensemble of well-chosen strong and diverse models

Ensemble models gain more accuracy and robustness by combining data from numerous modeling approaches.

Average

Regression ensemble prediction is calculated as the average of the member of predictions

$$P = \frac{p_1 + p_2 + p_3 + \cdots + p_n}{n}$$

In classification a class label prediction using mode of the member predictions.

The class probability is calculated as the argmax of the summed probabilities for each class label

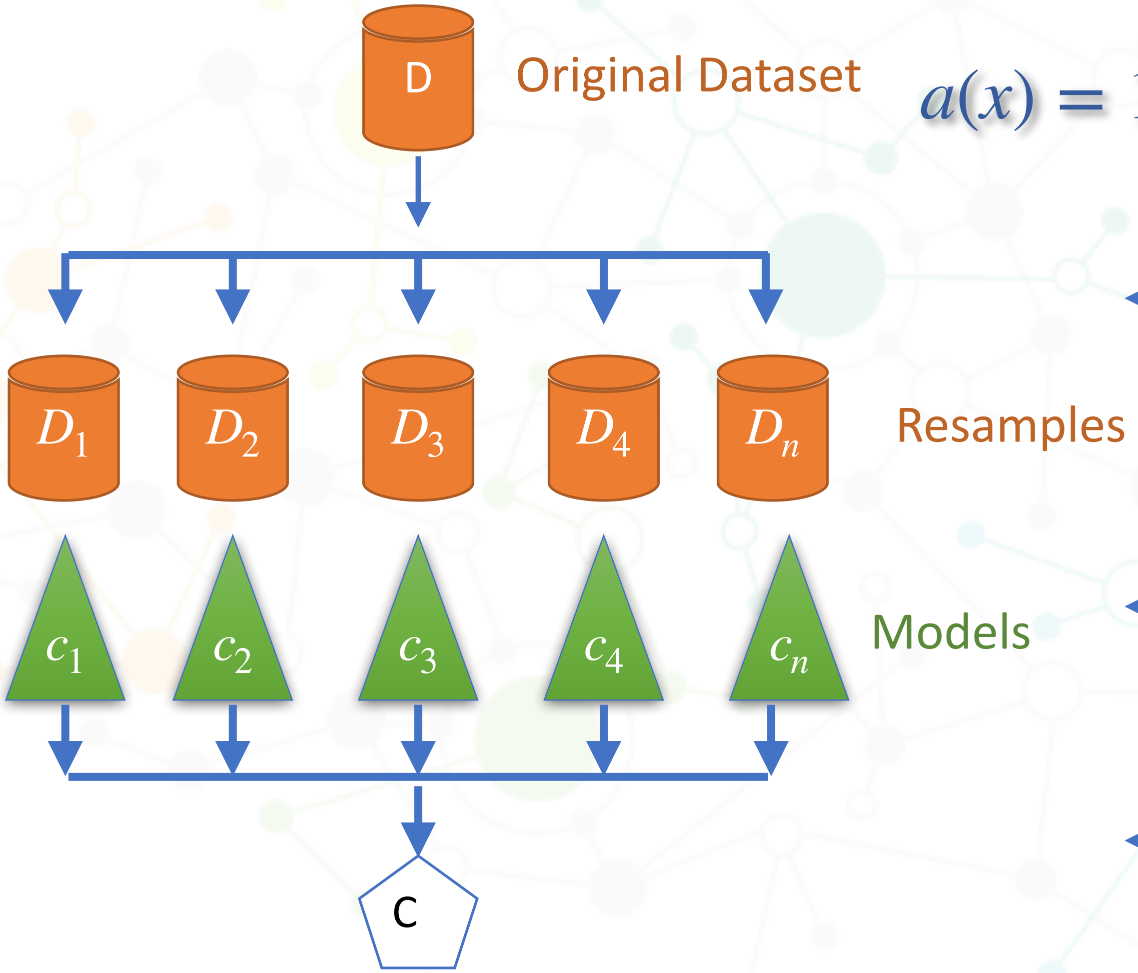
Bagging

- Bagging(Bootstrap Aggregating) is one of the first and most basic ensemble techniques.
- Bagging is based on the statistical method of bootstrapping which makes the evaluation of many statistics of complex models feasible.
- The bootstrap method goes as follows. Let there be a sample X of size N.
- All elements are equally likely to be selected, thus each element is drawn with the equal probability $\frac{1}{N}$.
- By repeating this procedure M times, we create M bootstrap samples $X_1, \ldots, X_M$. In the end, we have a sufficient number of samples and can compute various statistics of the original distribution.

Bagging

- **Bagging(bootstrapping) aggregate reduces variance of an estimate by taking mean of multiple estimates**

$$a(x) = 1/M \sum_{m=1}^{M} a_i(x)$$



Create random samples datasets of the original training dataset(bootstrapping)

Build and fit several models to each of these diverse copies

Take the average of all the predictions to make the final overall predictions

Bagging: Random Forest

- Random forest is a good example of bagging ensemble machine learning method
- Random forest technique combines various decision trees to produce a more generalized model
- random forests are utilized to produce de-correlated decision tree
- Smaller trees are built using these subsets creating tree diversity
- To overcome overfitting diverse set for decision tree are required

Boosting

- The term boosting refers to a family of algorithms which converts weak learner to strong learners.
- Boosting reduces a bias by training weak learners sequentially , each to correct its predecessor.
- Underlying engine used for boosting algorithms can be anything. There are many boosting algorithms which use other types of engine such:
    1. AdaBoosting(Adaptive boosting)
    2. Gradient Tree Boosting
    3. XGBoost

Boosting Algorithm

STEP 1

booting

- Train a classifier H_1 that best classifies the data with respect to accuracy

STEP 2

boosting

- Identify the regions where H1 produces errors, add weights to them and produce a H_2 classifier

STEP 3

boosting

- Aggregate those samples for which H_1 gives a different result from H_2 and producesH_3 classifiers
- Repeat step 2 for a new classifier

AdaBoost

- Boosting is a technique of changing weak learners into strong leaners.
- each new tree is a fit on a modified version of the original dataset
- AdaBoost is the first boosting algorithm to be adapted in solving practices.
- AdaBoost(discrete AdaBoost) is a specific boosting algorithm developed for a classification problems.
- It helps mixing multiple weak classifiers into one strong classifier
- The weakness is identified by the weak estimator’s error rate.

AdaBoost Algorithm

- A weak classifier is prepared on the training data using the weighted samples
- Only binary classification problems are supported
- Every decision stump makes one decision on one input variable and outputs
- The misclassification rate is calculated as:

$$error = \frac{correct - N}{N}$$

AdaBoost Algorithm

STEP 1

AdaBoost

- Initially each data point is weighted equally with weight $w_i = 1/n$

where n is the number of samples

STEP 2

AdaBoost

- A classifier H_1 is picked up that best classifies the data with minimal error rate

STEP 3

AdaBoost

- The weighing factor α is dependent on error(e_t) caused by the H_1 classifier

$$\alpha^t = \frac{1}{2}\ln\frac{1-e_t}{e_t}$$

STEP 4

AdaBoost

- Weight after time is given as:

$$\frac{w_t^{t+1}}{z}e^{-\alpha h_1(x).y(x)}$$

where z is the normalizing factor, $h_1(x)\,.\,y(x)$ is sign of the current output

AdaBoost Flowchart

AdaBoost selects a training subset randomly

↓

It iteratively trains the AdaBoost ML model

↓

It assigns a higher weight to wrongly classified observations

↓

It assigns weights to the trained classifier in each iterations according to the accuracy of the classifier

Gradient Boosting(GBM)

- Gradient boosting trains several models in a very gradual, additive, and sequential manner
- GDM minimizes the loss function(MSE) of a model by adding weak learners using a gradient descent procedure
- Gradient boosting involves three elements:
  1. a loss function to be optimized
  2. a weak learner to make predictions
  3. an additive models to add weak learners to minimize the loss function

XGBoost

- eXtreme Gradient Boosting is a library for developing fast and high performance gradient boosting tree models
- A more regularized model to control over-fitting and give better performance
- Tree-based algorithm for:
  - classification
  - Regression
  - Ranking

  with custom loss functions
- Used extensively in ML competitions as it is ten times faster than other gradient based techniques.

XGBoosting Algorithm

1

SYSTEM

- Parallelization
  Tree construction using all CPU cores while training
- Distributed Computing
  Training very large models using a cluster
- Cache Optimization
  data structure make best use of hardwares

2

ALGORITHM

- **Spares Aware**
  automatic handing of missing data values
- **Block Structure**
  supports the parallelization of tree
- **Continues Training**
  to boost an already cited model on new

3

MODELS

- **Gradient Boosting**
- **Stochastic Gradient Boosting**
- **Regularized Gradient Boosting**

Stacking

- Stacking is a heterogeneous ensemble method the combine different types of classifiers in which each classifier built upon the same data.
- This technique works by allowing a training algorithm to ensemble several other similar learning algorithm predictions.
- The overall result of stacking method is carried out by averaging all the result of each combined model
- Unlike boosting each lower-level model is undergone into parallel training.
- The top layer model has good prediction accuracy and they built based on lower-level models.
- The stack goes on increasing until the best prediction is carried out with a minimum error

THANK YOU!