Applied Data Science

Session 27: Ensemble: Stacking & Boosting

Dr. Soharab Hossain Shaikh

1

Ensemble Methods: Types

- **Bagging:** train learners in parallel on different samples of the data, then combine by voting (discrete output) or by averaging (continuous output).
- **Stacking:** combine model outputs using a second-stage learner like linear regression (different types of learners).
- **Boosting:** train learners on the filtered output of other learners (sequential).

2

Bagging Issues

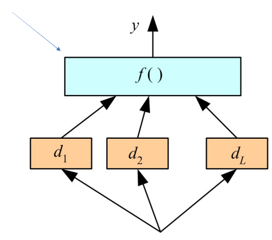
1. Homogeneous Learners: Similar classifiers usually make similar errors - so forming an ensemble with similar classifiers may not improve the classification accuracy.
2. At the presence of a classifier that performs much better than all available base classifiers, may cause degradation in the overall performance.
3. A poorly performing classifier may cause performance deterioration of the ensemble.
4. Amount of correlation among the incorrect classifications made by the base classifiers.
5. Dominance of a classifier: If the consistent classifiers tend to misclassify the same instances, then combining their results will have no benefit.
6. Diversity of classifiers – Heterogeneous Learners: In contrast, a greater degree of independence among the classifiers can result in errors made by individual classifiers being overlooked when the results of the ensemble are combined.

3

Stacking

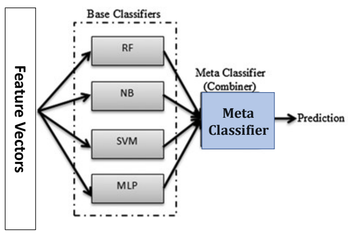
Combiner/Meta-Learner $f()$ is another learner.

- Uses *meta learner (level-1)* instead of voting to combine predictions of base (level-0) learners
- Predictions of base learners *(level-0 models)* are used as input for meta learner *(level-1 model).*
- Base learners usually different learning schemes (heterogeneous models).
- Hard to analyze theoretically: “black magic”



4

Stacking

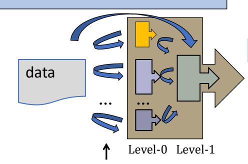


1. Train several different models (base/level-0 models) on the original data
2. Train a new model (meta classifier/ level-1) based on the outputs of level-0 models.

5

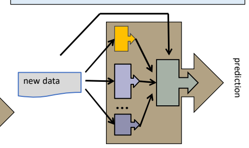
Stacking

1. Train different models on the same data ("Level-0 models")
2. Train a new ("Level-1") model with the outputs of the Level-0 models



Using different parameters and/or different data mining techniques

2. Given a new unlabeled data instance, input it to each Level-0 model:



The ensemble prediction is the Level-1 model prediction based on the Level-0 model predictions

May help reduce prediction error

6

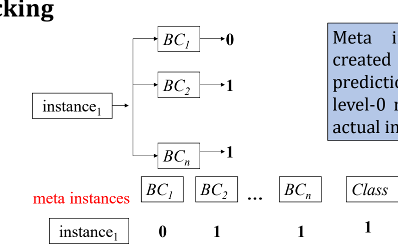
Stacking

Algorithm Stacking

Input: training data $D = \{x_i, y_i\}_{i=1}^m$
Ouput: ensemble classifier H
Step 1: learn base-level classifiers
for $t = 1$ to T **do**
  learn h_t based on D
end for
Step 2: construct new data set of predictions
for $i = 1$ to m **do**
  $D_h = \{x'_i, y_i\}$, where $x'_i = \{h_1(x_i), ..., h_T(x_i)\}$
end for
Step 3: learn a meta-classifier
learn H based on D_h
return H

7

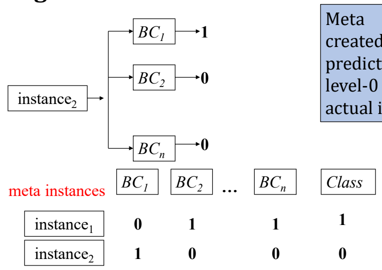
Stacking



Meta instances are created from the predictions of the level-0 models on the actual instances.

8

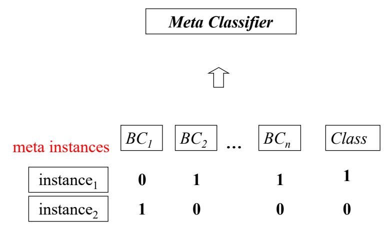
Stacking



Meta instances are created from the predictions of the level-0 models on the actual instances.

9

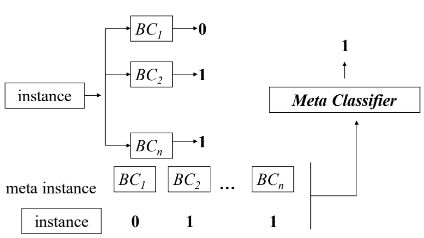
Stacking



Training:
Meta classifier (level 1) is trained on the meta instances.

10

Stacking



Testing:
New instance is passed through all the level-0 models to get their individual prediction – and an instance for the level-1 model is created from that.

This instance is fed to the level-1 model in order to get the final prediction.

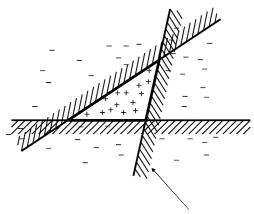
11

Boosting Intuition

- Train a number of weak classifiers (e.g. decision trees) in a sequence.
- We adaptively weigh each data case.
- Data cases which are wrongly classified get high weight.
- A new classifier should focus on those cases which were incorrectly classified in the last round.
- Each boosting round learns a new (simple/weak) classifier on the weighted dataset.
- These weak classifiers are weighted to combine them into a single powerful/strong classifier. Combine the classifiers by letting them vote on the final prediction (like bagging).
- Classifiers that that obtain low training error rate have high weight.
- We stop by using monitoring a hold out set (cross-validation).

12

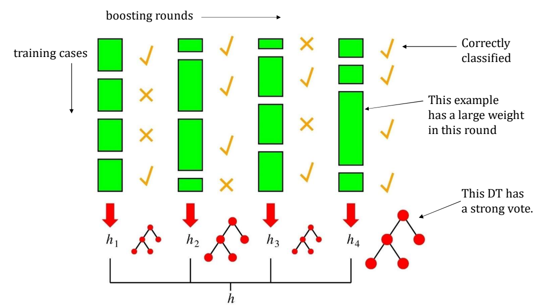
Example



This line is one simple classifier saying that everything to the left is + and everything to the right is -

13

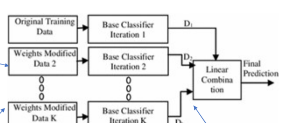
Boosting Process



14

Adaboost

Training data from base data with focus on instances which were incorrectly classified by earlier model (if any)



K similar classifiers created in sequence with respective training data with focus on addressing the misclassified data, not the usual cost functions

Voting could be simple or weighted

It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights to incorrectly classified instance.

15

Adaboost

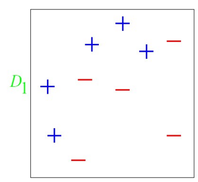
Algorithm AdaBoost

Init data weights $\{w_n\}$ to $1/N$
for $m = 1$ to M **do**
  fit a classifier $y_m(x)$ by minimizing weighted error function J_m:
  $J_m = \sum_{n=1}^{N} w_n^{(m)} 1[y_m(x_n) \neq t_n]$
  compute $\epsilon_m = \sum_{n=1}^{N} w_n^{(m)} 1[y_m(x_n) \neq t_n] / \sum_{n=1}^{N} w_n^{(m)}$
  evaluate $\alpha_m = \log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$
  update the data weights: $w_n^{(m+1)} = w_n^{(m)} \exp\{\alpha_m 1[y_m(x_n) \neq t_n]\}$
end for
Make predictions using the final model: $Y_M(x) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(x)\right)$

AdaBoost, short for *Adaptive Boosting*, is a machine learning meta-algorithm formulated by Yoav Freund and Robert Schapire, who won the 2003 Gödel Prize for their work.

16

Adaboost

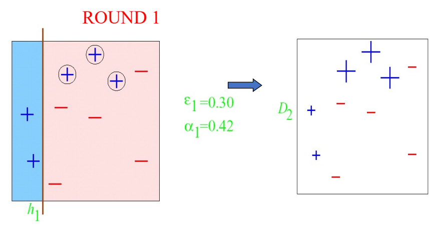


Original training set: equal weights to all training samples

17

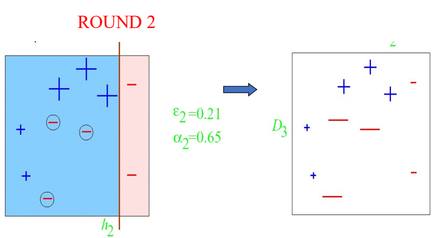
Adaboost

ε = error rate of classifier
α = weight of classifier



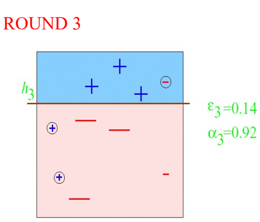
18

Adaboost



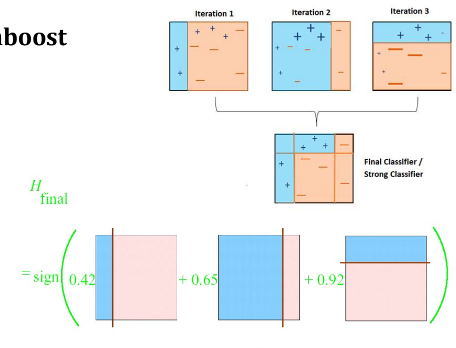
19

Adaboost



20

Adaboost



21

How to Train each Classifier?

input : $\mathbf{x}$, output : $y(\mathbf{x}) \in \{1, -1\}$

target : $t \in \{1, -1\}$,

weight on case n for classifier m : w_n^m

Cost function for classifier m :

$$J_m = \sum_{n=1}^{N} w_n^m \left[y_m(\mathbf{x}_n) \neq t_n \right] = \sum weighted\ errors$$

1 if error,
0 if correct

22

How to weigh each training case for Classifier *m*?

$$Let \quad \varepsilon_m = \frac{J_m}{\sum_n w_n^m}$$

weighted error rate of classifier

$$Let \quad \alpha_m = \ln\left(\frac{1-\varepsilon_m}{\varepsilon_m}\right)$$

This is the quality of the classifier. It is zero if the classifier has weighted error rate of 0.5 and infinity if the classifier is perfect

$$w_n^{m+1} = w_n^m \exp\{ \alpha_m [y_m(x_n) \neq t_n] \}$$

23

How to weigh each training case for Classifier *m*?

- Weight the binary prediction of each classifier by the quality of that classifier:

$$Y_M(\mathbf{x}) = sign\left(\sum_{m=1}^{M} \alpha_m \, y_m(\mathbf{x}) \right)$$

24

Adaboost Algorithm

- Initialize the data weights w_n = 1/N.
- For m=1,..,M:
  - Fit a classifier $y_m(\mathbf{x})$ to the training data by minimizing the weighted error function:

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I(y_m(\mathbf{x}_n) \neq t_n),$$

where $\boxed{I(y_m(\mathbf{x}_n) \neq t_n)}$ is the indicator function and equals to one when $y_m(\mathbf{x}_n) \neq t_n$ and zero otherwise.

  - Evaluate:

$$\alpha_m = \ln \frac{1-\epsilon_m}{\epsilon_m}, \quad \epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(y_m(\mathbf{x}_n) \neq t_n)}{\sum_{n=1}^{N} w_n^{(m)}}.$$

Weighting coefficients.

weighted measures of the error rates.

25

Adaboost Algorithm

- Initialize the data weights w_n = 1/N.
- For m=1,..,M:
  - Fit a classifier $y_m(\mathbf{x})$ to the training data by minimizing:

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I(y_m(\mathbf{x}_n) \neq t_n),$$

  - Evaluate:

$$\alpha_m = \ln \frac{1-\epsilon_m}{\epsilon_m}, \quad \epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(y_m(\mathbf{x}_n) \neq t_n)}{\sum_{n=1}^{N} w_n^{(m)}}.$$

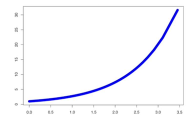
  - Update the data weights:

$$w_n^{(m+1)} = w_n^{(m)} \exp\left(\alpha_m I(y_m(\mathbf{x}_n) \neq t_n)\right).$$

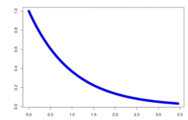
- Make predictions using the final model:

$$Y_M(\mathbf{x}) = \operatorname{sign}\left(\sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})\right).$$

Exponential Loss



Positive Exponent



Negative Exponent

26

27

Case-study

28

Classification of Credit Risk - Predicting Possible Credit Defaulter based on German Credit Dataset

Context
The original dataset contains 1000 entries with 20 categorial/symbolic attributes prepared by Prof. Hofmann.

In this dataset, each entry represents a person who takes a credit by a bank. ach person is class ified as good or bad credit risks according to the set of attributes.

Description of the Attributes

> Age (numeric)

> Sex (text: male, female)

> Job (numeric: 0 - unskilled and non-resident, 1 - unskilled and resident, 2 - skilled, 3 - highly skilled)

> Housing (text: own, rent, or free)

> Saving accounts (text - little, moderate, quite rich, rich)

> Checking account (numeric, in DM - Deutsch Mark)

> Credit amount (numeric, in DM)

> Duration (numeric, in month)

> Purpose (text: car, furniture/equipment, radio/TV, domestic appliances, repairs, education, b usiness, vacation/others) etc.....

Total of 17 features (16 input features and 1 target variable (*'default'* column))

29

Let’s go to the Coding Demo…

30

To be continued in the next session…..

31