COMS30035, Machine learning: Combining Models 2, Ensembles

Edwin Simpson
edwin.simpson@bristol.ac.uk

Department of Computer Science, SCEEM
University of Bristol

November 14, 2023

Agenda

- Model Selection
- Model Averaging
- Ensembles: Bagging
- Ensembles: Boosting and Stacking
- Tree-based Models
- Conditional Mixture Models
- Ensembles of Humans

Bayesian Model Averaging (BMA)

- Model selection does not always pick out one model, h, as a clear winner
- We may be *uncertain* about which model is correct

Bayesian Model Averaging (BMA)

- Model selection does not always pick out one model, h, as a clear winner
- We may be *uncertain* about which model is correct
- We can express uncertainty by assigning a probability to each model given the training data, $p(h|\boldsymbol{X})$.

Bayesian Model Averaging (BMA)

- Rather than choosing a single model, we can now take an expectation.
- Our predictions now come from a *weighted sum* over models, where $p(h|\boldsymbol{X})$ are weights :

$$p(\boldsymbol{z}|\boldsymbol{X}) = \sum_{h=1}^{H} p(\boldsymbol{z}|\boldsymbol{X}, h) p(h|\boldsymbol{X}) \quad (1)$$

Bayesian Model Averaging (BMA)

- Apply Bayes' rule to estimate the weights:

$$p(h|\boldsymbol{X}) = \frac{p(\boldsymbol{X}|h)p(h)}{\sum_{h'=1}^{H} p(\boldsymbol{X}|h')p(h')} \tag{2}$$

- What happens as we increase the amount of data in $\boldsymbol{X}$?

Bayesian Model Averaging (BMA)

- Apply Bayes' rule to estimate the weights:

$$p(h|\boldsymbol{X}) = \frac{p(\boldsymbol{X}|h)p(h)}{\sum_{h'=1}^{H} p(\boldsymbol{X}|h')p(h')} \tag{2}$$

- What happens as we increase the amount of data in $\boldsymbol{X}$?
- $p(h|\boldsymbol{X})$ becomes more focussed on one model.
- So BMA is soft model selection, it does not *combine* models to make a more powerful model.

Ensemble Methods

- Model Selection
- Model Averaging
- Ensembles: Bagging
- Ensembles: Boosting and Stacking
- Tree-based Models
- Conditional Mixture Models
- Ensembles of Humans

Wisdom of the crowd

Guess the weight!

Wisdom of the crowd

Guess the weight!
In 1907, Sir Francis Galton asked 787 villagers to guess the weight of an ox. None of them got the right answer, but when Galton averaged their guesses, he arrived at a near perfect estimate.

Wisdom of the crowd

Guess the weight!
In 1907, Sir Francis Galton asked 787 villagers to guess the weight of an ox. None of them got the right answer, but when Galton averaged their guesses, he arrived at a near perfect estimate.
The combination was more effective than any one ‘model’.

Ensemble Methods

- Ensemble: a combination of different models.
- Often outperforms the average individual, and sometimes even the best individual.
- Different principle to BMA:
  - BMA weights try to identify a single, correct model
  - BMA weights do not provide the optimal combination

Expected Error of an Ensemble

- Given a set of models, $1, ..., M$,
- $y_m(\boldsymbol{x})$ is the prediction from model m.
- Simple ensemble: the mean of the individual predictions, $y_{COM} = \frac{1}{M} \sum_{m=1}^{M} y_m(\boldsymbol{x})$,

Expected Error of an Ensemble

- Given a set of models, $1, ..., M$,
- $y_m(\boldsymbol{x})$ is the prediction from model m.
- Simple ensemble: the mean of the individual predictions, $y_{COM} = \frac{1}{M}\sum_{m=1}^{M} y_m(\boldsymbol{x})$,
- Let's compare the sum-of-squares error of y_{COM} with that of the individual models...

Expected Error of an Ensemble

Firstly, the error of our combination for a particular input $\boldsymbol{x}$ is:

$$(y(\boldsymbol{x}) - y_{COM}(\boldsymbol{x}))^2 = \left(\frac{1}{M} \sum_{m=1}^{M} (y(\boldsymbol{x}) - y_m(\boldsymbol{x})) \right)^2 . \tag{3}$$

Expected Error of an Ensemble

Firstly, the expected error of our combination is:

$$E_{COM} = \mathbb{E}_{\boldsymbol{x}}[(y(\boldsymbol{x}) - y_{COM}(\boldsymbol{x}))^2] = \mathbb{E}_{\boldsymbol{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))\right)^2\right]. \quad (4)$$

Expected Error of an Ensemble

- The expected error of an individual model m is: $E_m = \mathbb{E}_{\boldsymbol{x}}[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2]$.

Expected Error of an Ensemble

- The **average** expected error of an individual model is:
  $E_{AV} = \frac{1}{M} \sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}} \left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2 \right]$.

Expected Error of an Ensemble

- The **average** expected error of an individual model is: $E_{AV} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}}\left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2\right]$.
- Remember: $E_{COM} = \mathbb{E}_{\boldsymbol{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))\right)^2\right]$.

Expected Error of an Ensemble

- The **average** expected error of an individual model is: $E_{AV} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}}\left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2\right]$.
- Remember: $E_{COM} = \mathbb{E}_{\boldsymbol{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))\right)^2\right]$.
- ...so we have $E_{COM} = \frac{1}{M}E_{AV}$, since for E_{COM}, the $\frac{1}{M}$ is squared

Expected Error of an Ensemble

- The **average** expected error of an individual model is: $E_{AV} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}}\left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2\right]$.
- Remember: $E_{COM} = \mathbb{E}_{\boldsymbol{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))\right)^2\right]$.
- ...so we have $E_{COM} = \frac{1}{M}E_{AV}$, since for E_{COM}, the $\frac{1}{M}$ is squared
- This relies on two assumptions...
  1. The errors of each model have zero mean;
  2. The errors of different models are not correlated;

Edwin Simpson
edwin.simpson@bristol.ac.uk

Expected Error of an Ensemble

- The **average** expected error of an individual model is: $E_{AV} = \frac{1}{M} \sum_{m=1}^{M} \mathbb{E}_{\boldsymbol{x}} \left[(y(\boldsymbol{x}) - y_m(\boldsymbol{x}))^2 \right]$.
- Remember: $E_{COM} = \mathbb{E}_{\boldsymbol{x}} \left[\left(\frac{1}{M} \sum_{m=1}^{M} (y(\boldsymbol{x}) - y_m(\boldsymbol{x})) \right)^2 \right]$.
- ...so we have $E_{COM} = \frac{1}{M} E_{AV}$, since for E_{COM}, the $\frac{1}{M}$ is squared
- This relies on two assumptions...
  1. The errors of each model have zero mean;
  2. The errors of different models are not correlated;
- Intuition: if models make different, random errors, they will tend to cancel out.

Expected Error of an Ensemble

- $E_{COM} = \frac{1}{M} E_{AV}$ is pretty amazing, but is it realistic?

[1] This bound is due to *Jensen's inequality*.

Expected Error of an Ensemble

- $E_{COM} = \frac{1}{M} E_{AV}$ is pretty amazing, but is it realistic?
- No, because we have made extreme assumptions about the models' errors – in practice, they are usually highly correlated and biased.

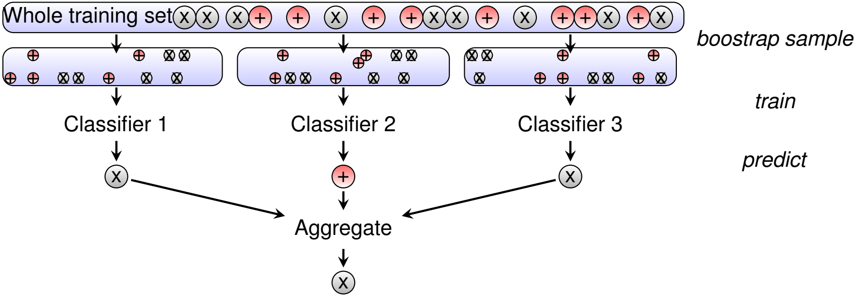
[1] This bound is due to *Jensen's inequality*.

Expected Error of an Ensemble

- $E_{COM} = \frac{1}{M} E_{AV}$ is pretty amazing, but is it realistic?
- No, because we have made extreme assumptions about the models' errors – in practice, they are usually highly correlated and biased.
- However, the combined error cannot be worse than the average error: $E_{COM} \leq E_{AV}$[1]
- The results tells us that the models should be *diverse* to avoid repeating the same errors.

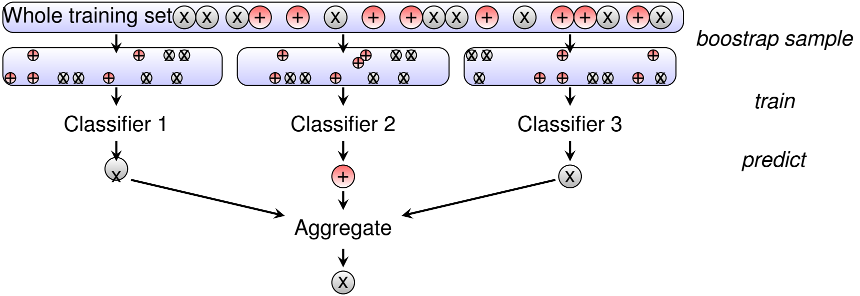
[1] This bound is due to *Jensen's inequality*.

Bootstrap Aggregation (Bagging)



- Create diversity by training models on different samples of the training set.

Bootstrap Aggregation (Bagging)



- Create diversity by training models on different samples of the training set.
- For each model *m*, randomly sample *N* data points with replacement from a training set with *N* data points and train *m* on the subsample.
- In each sample, some data points will be repeated and others will be omitted.
- Combine predictions by taking the mean or majority vote.

Boosting

- Can we do better than choosing training sets at random?

Boosting

- Can we do better than choosing training sets at random?
- Train *base* models sequentially, ensuring that each base model addresses the weaknesses of the ensemble.
- Instead random sampling, weight the data points in the training set according to the performance of previous base models.

Boosting

- Can we do better than choosing training sets at random?
- Train *base* models sequentially, ensuring that each base model addresses the weaknesses of the ensemble.
- Instead random sampling, weight the data points in the training set according to the performance of previous base models.
- *AdaBoost* is a popular boosting method originally designed for *binary classification*.

AdaBoost

Training sequence $\rightarrow$

train new classifier on weighted data that outputs class labels $+1$ or -1

$$\{w_n^{(1)}\}$$

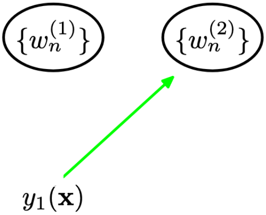
$$\downarrow$$

$$y_1(\mathbf{x})$$

Edwin Simpson
edwin.simpson@bristol.ac.uk

AdaBoost

Training sequence $\rightarrow$

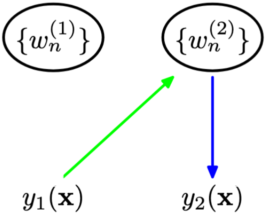


compute weights from performance of previous classifier

Edwin Simpson
edwin.simpson@bristol.ac.uk

AdaBoost

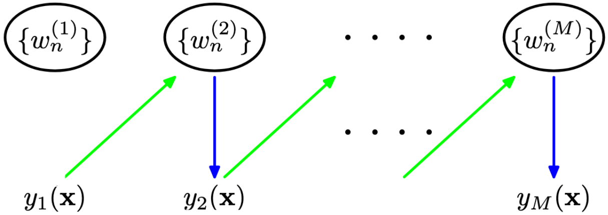
Training sequence $\rightarrow$



compute weights from performance of previous classifier

AdaBoost

Training sequence $\rightarrow$

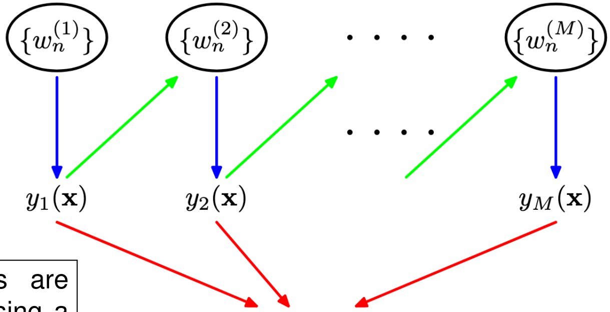


compute weights from performance of previous classifier

$$\mathrm{n}\left(\sum_{m}^{M} \alpha_m y_m(\mathbf{x})\right)$$

AdaBoost

Training sequence $\rightarrow$



after all classifiers are trained, combine using a weighted sum

$$Y_M(\mathbf{x}) = \text{sign}\left(\sum_m^M \alpha_m y_m(\mathbf{x})\right)$$

AdaBoost: Data Weights

1. Initialise $w_n^{(1)} = \frac{1}{N}$ for all data points n;

AdaBoost: Data Weights

1. Initialise $w_n^{(1)} = \frac{1}{N}$ for all data points n;
2. Compute the weighted error rate of model m:

$$\epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} [y_m(\boldsymbol{x}_n) \neq y(\boldsymbol{x}_n)]}{\sum_{n=1}^{N} w_n^{(m)}} \tag{5}$$

AdaBoost: Data Weights

1. Initialise $w_n^{(1)} = \frac{1}{N}$ for all data points n;
2. Compute the weighted error rate of model m:

$$\epsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} [y_m(\boldsymbol{x}_n) \neq y(\boldsymbol{x}_n)]}{\sum_{n=1}^{N} w_n^{(m)}} \tag{5}$$

3. Update the weight for each data point n:

$$w_n^{(m+1)} = \begin{cases} w_n^{(m)} \left(\frac{1-\epsilon_m}{\epsilon_m}\right) & \text{if } y_m(\boldsymbol{x}_n) \neq y(\boldsymbol{x}_n) \\ w_n^{(m)} & \text{if } y_m(\boldsymbol{x}_n) = y(\boldsymbol{x}_n) \end{cases} \tag{6}$$

- The weight is increased when m makes an incorrect prediction.

AdaBoost: Final Classifier Weights

$$y_M(\boldsymbol{x}_n) = \sum_{m=1}^{M} \alpha_m y_m(\boldsymbol{x}_n) \tag{7}$$

AdaBoost: Final Classifier Weights

$$y_M(\boldsymbol{x}_n) = \sum_{m=1}^{M} \alpha_m y_m(\boldsymbol{x}_n) \tag{7}$$

- Choosing the weight α_m for m that minimises the exponential loss of m gives us:

$$\alpha_m = \ln\left\{\frac{1-\epsilon_m}{\epsilon_m}\right\} \tag{8}$$

AdaBoost: Final Classifier Weights

$$y_M(\boldsymbol{x}_n) = \sum_{m=1}^{M} \alpha_m y_m(\boldsymbol{x}_n) \tag{7}$$

- Choosing the weight α_m for m that minimises the exponential loss of m gives us:

$$\alpha_m = \ln\left\{\frac{1-\epsilon_m}{\epsilon_m}\right\} \tag{8}$$

- Weights are higher for classifiers with a lower error rate.

AdaBoost: Final Classifier Weights

$$y_M(\boldsymbol{x}_n) = \sum_{m=1}^{M} \alpha_m y_m(\boldsymbol{x}_n) \tag{7}$$

- Choosing the weight α_m for m that minimises the exponential loss of m gives us:

$$\alpha_m = \ln \left\{ \frac{1 - \epsilon_m}{\epsilon_m} \right\} \tag{8}$$

- Weights are higher for classifiers with a lower error rate.
- Note that α_m is a log-odds function: AdaBoost optimises the approximation to the log-odds ratio.
- Other loss functions can be used to derive similar boosting schemes for regression and multi-class classification.

Stacking

Given a set of trained base classifiers, what's the best way to combine them?

Stacking

- Given a trained set of base classifiers, learn the combination function!

Stacking

- Given a trained set of base classifiers, learn the combination function!
- Bagging: the combination function was majority vote, which is *unweighted*;

Stacking

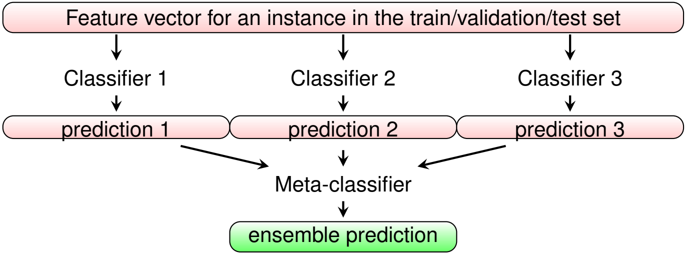
- Given a trained set of base classifiers, learn the combination function!
- Bagging: the combination function was majority vote, which is *unweighted*;
- Adaboost: a *weighted* sum of classifier outputs determined by model error rates;

Stacking

- Given a trained set of base classifiers, learn the combination function!
- Bagging: the combination function was majority vote, which is *unweighted*;
- Adaboost: a *weighted* sum of classifier outputs determined by model error rates;
- Stacking: use another classifier/regressor to learn a combination function that minimises the error rate of the entire ensemble

Stacking

- Given a trained set of base classifiers, learn the combination function!
- Bagging: the combination function was majority vote, which is *unweighted*;
- Adaboost: a *weighted* sum of classifier outputs determined by model error rates;
- Stacking: use another classifier/regressor to learn a combination function that minimises the error rate of the entire ensemble



Now do the quiz!

Please do the quiz for this lecture on Blackboard.