QBUS6850 Week 8

Ensemble Methods 2 - Boosting

Dr Stephen Tierney

The University of Sydney Business School

The Plan

- Part 1 (Week 6): Understand decision trees and How to make an ensemble (forests)
- **Part 2 (Week 8): Advanced ensembles 1 (boosting)**
- Part 3 (Week 9): Advanced ensembles 2 (gradient boosting)

Reading

- Chapter 10.1 - 10.5, The Elements of Statistical Learning, Hastie et al.

Bagging

Bagging aggregates individual weak models by assigning an equal weight to each model.

Boosting

Boosting aggregates many weak models by:

- successively adding them to the overall model
- assigning different weights to each weak model

Boosting

When we add another weak model it should correct the shortcomings of the overall model.

Boosting

Boosting in general does not require bootstrapping.

¿Por qué no los dos?

Boosting is a meta-learning algorithm. It learns about a set of learners. This means that boosting is agnostic to the type of weak learner in the ensemble.

As such they can be used for both **regression** and **classification** tasks!

In these slides we will discuss a particular boosting algorithm and task: **AdaBoost for classification**.

Simplified AdaBoost

Simplified AdaBoost Algorithm

1. Train a number of bootstrapped weak classifiers e.g. decision stumps
2. Iteratively refine the voting power of each classifier:
   - 2.1 Pick the classifier with lowest weighted misclassification rate
   - 2.2 Calculate the classifier's "voting power"
   - 2.3 Update weights of all data points

There are two sets of parameters that we will be operating on: **data weights** and **classifier weights**.

AdaBoost Predictions

The final classification is generated by a weighted vote.

Classifiers that have low training misclassification/error rates have high voting power.

Simplified AdaBoost - Intuition

This simplified version of AdaBoost works similar to how you might study using past and practice exam papers.

Repeating:

- Attempt the exam paper
- Check your answers
- Study the questions/topics you got wrong

Each time you are correcting the mistakes that you previously made.

AdaBoost Example - Setup

Consider a dataset
$\mathcal{D} = \{A(x_{11}, y_1), B(x_{21}, y_2), C(x_{31}, y_3), D(x_{41}, y_4), E(x_{51}, y_5)\}$
where the positive cases are A, B, D and E, while the negative case is C.

There are $N = 5$ cases and 1 feature ($d = 1$).

Data	x	y
A	1.5	1
B	1.5	1
C	3.0	0
D	7.0	1
E	7.0	1

AdaBoost Example - Setup

Suppose we have already trained 6 weak classifiers, which are decision stumps with the following rules

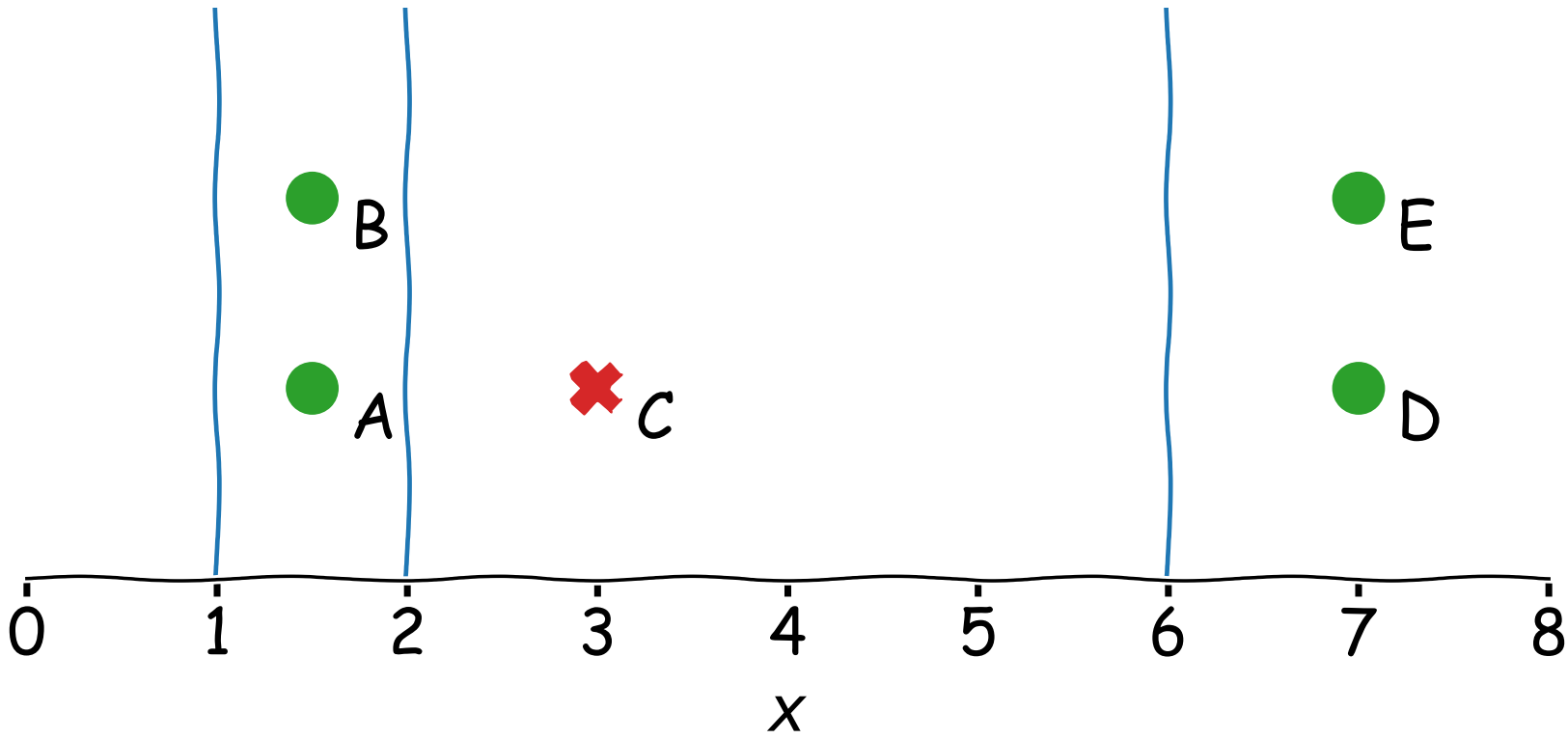
Classifiers
$x < 1$
$x < 2$
$x < 6$
$x \geq 1$
$x \geq 2$
$x \geq 6$

If condition of decision stump is satisfied:

- then predict 1 (positive)
- otherwise predict -1 (negative)

AdaBoost Example - Setup

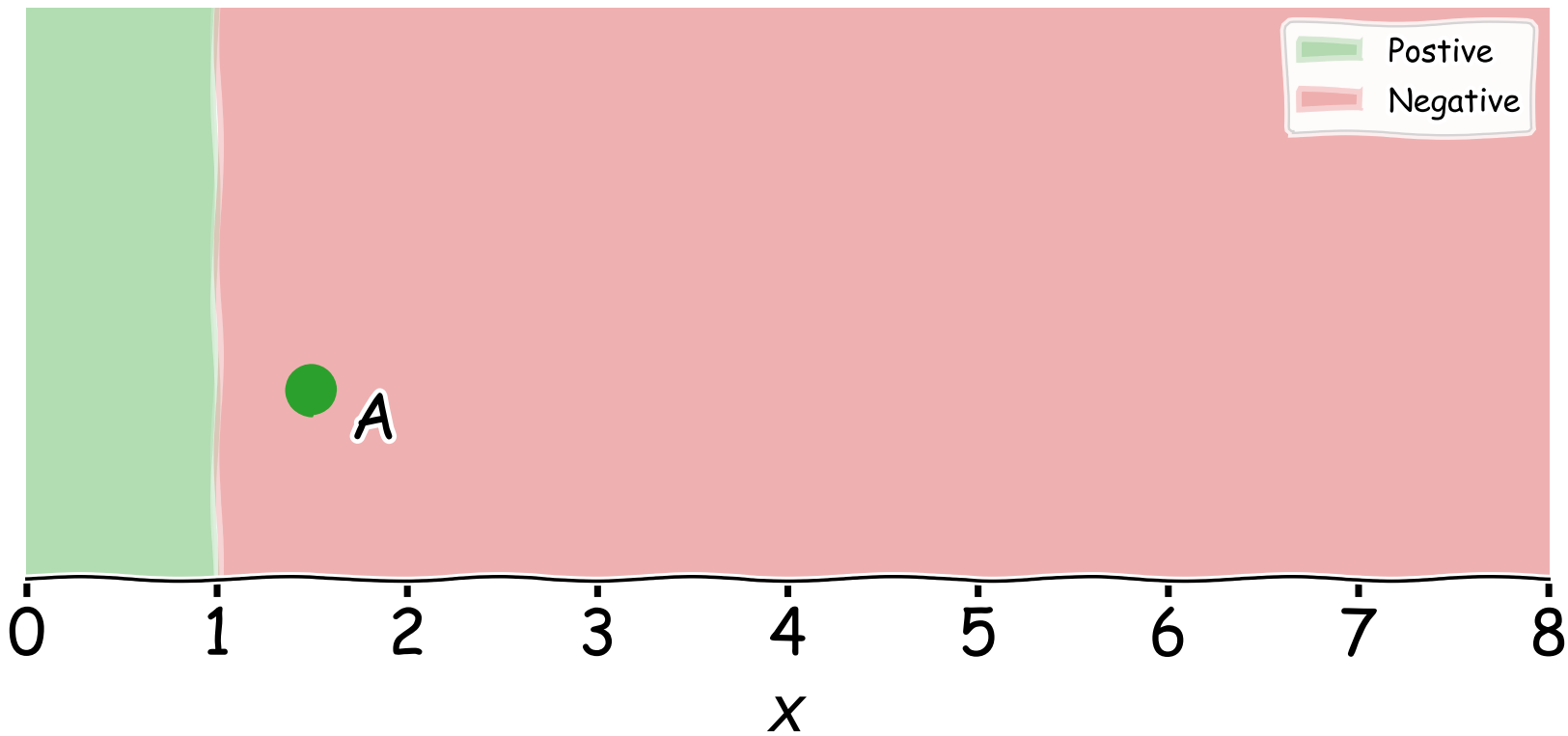
We can overlay our decision boundaries on the data points.



AdaBoost Example - Stump Demo

Consider the first classifier $x < 1$ and the case A.

The input x of case A does not satisfy the condition $x_{11} < 1$, hence the classifier will classify it as negative, which is wrong.



AdaBoost Example - Initialisation

Initialise data weights to be equal

$$w_i = \frac{1}{N} = \frac{1}{5}; \quad i = 1, 2, 3, 4, 5$$

Data	Weight Iter 1
A	1/5
B	1/5
C	1/5
D	1/5
E	1/5

AdaBoost Example - Iteration 1

1.1 Calculate misclassification rate ϵ_k for classifiers $\widehat{F}_k(x)$

Classifiers	Misclassified	Rate
$x < 1$	A, B, D, E	1/5+1/5+1/5+1/5 = 4/5
$x < 2$	D, E	1/5+1/5 = 2/5
$x < 6$	C, D, E	1/5+1/5+1/5 = 3/5
$x \geq 1$	C	1/5 = 1/5
$x \geq 2$	A, B, C	1/5+1/5+1/5 = 3/5
$x \geq 6$	A, B	1/5+1/5 = 2/5

where $\epsilon_k = \sum_i^N w_i(\widehat{F}_k(x_i) \neq y_i)$

AdaBoost Example - Iteration 1

1.2 Pick the $\widehat{F}_k(x)$ with the lowest error, i.e. $\widehat{F}_4(x)$.

Intuitively we pick the best classifier as our starting point because it means we have to fix the least amount of mistakes.

1.3 Calculate voting power for the best classifier. AdaBoost uses natural log as the voting power

$$\alpha_k = \frac{1}{2}\log\left(\frac{1-\epsilon_k}{\epsilon_k}\right) = \frac{1}{2}\log\left(\frac{1-\frac{1}{5}}{\frac{1}{5}}\right) = \frac{1}{2}\log(4)$$

The lower misclassification rate, the higher voting power. In other words, if a classifier is accurate, then we trust it more.

AdaBoost Example - Iteration 1

1.4 Update the ensemble model

$$\overline{F}(\mathrm{x}, \beta) = \frac{1}{2} \log(4) \widehat{F}_4(\mathrm{x}).$$

AdaBoost Example - Iteration 1

Our ensemble so far consists of a single classifier

$$\overline{F}(\mathrm{x}, \beta) = \frac{1}{2}\log(4)\widehat{F}_4(\mathrm{x}).$$

In the next iterations we need to focus on the data points that we misclassify.

To do this we can apply a larger weight to the misclassified points.

In this example we want to give C a relatively large weight.

AdaBoost Example - Iteration 1

1.5 Update the data weights using ϵ of the most recent classifier

$$w_i^{new} = \begin{cases} \frac{1}{2(1-\epsilon)} w_i^{old} & \text{if the case is correct} \\ \frac{1}{2\epsilon} w_i^{old} & \text{if the case is incorrect} \end{cases} \quad (1)$$

Since case C is incorrectly classified by $\widehat{F}_4(x)$, the new weight for case C is

$$w_3^{new} = \frac{1}{2\epsilon} w_3^{old} = \frac{1}{2 \times (1/5)} \frac{1}{5} = \frac{1}{2}.$$

We have calculated all the new weights for the five cases

Data	Weight Iter 1	Weight Iter 2
A	1/5	1/8
B	1/5	1/8
C	1/5	1/2
D	1/5	1/8
E	1/5	1/8

Note: the new weights satisfy the following condition $\sum_{correct} w_i = \sum_{incorrect} w_i = \frac{1}{2}$.

AdaBoost Example - Iteration 1

1.6 Stop if one of the following conditions is met

1. Ensemble $\overline{F}(\mathrm{x})$ has perfect classification
2. Maximum iterations reached
3. No good classifier left, e.g., the best unused classifier has misclassification rate 0.5

AdaBoost Example - Iteration 2

2.1 Calculate misclassification rate ϵ_k for each classifier $\widehat{F}_k(x)$.

2.2 Pick $\widehat{F}_k(x)$ with the lowest misclassification rate. If there is a draw e.g. $\widehat{F}_2$ and $\widehat{F}_6$, arbitrarily select one. In this example, we pick $\widehat{F}_2(x)$

Data	Weight Iter 1	Weight Iter 2
A	1/5	1/8
B	1/5	1/8
C	1/5	1/2
D	1/5	1/8
E	1/5	1/8

Classifiers	Misclassified	Rate
$x < 1$	A, B, D, E	1/8+1/8+1/8+1/8 = 1/2
$x < 2$	D, E	1/8+1/8 = 1/4
$x < 6$	C, D, E	1/2+1/8+1/8 = 3/4
$x \geq 1$	C	1/2 = 1/2
$x \geq 2$	A, B, C	1/8+1/8+1/2 = 3/4
$x \geq 6$	A, B	1/8+1/8 = 1/4

AdaBoost Example - Iteration 2

2.3 Calculate voting power for the best classifier $\widehat{F}_2(\mathrm{x}, \beta^{(2)})$.

$$\alpha_k = \frac{1}{2}\log\left(\frac{1-\epsilon_k}{\epsilon_k}\right) = \frac{1}{2}\log(3)$$

and construct the current best classifier

$$\frac{1}{2}\log(4)\widehat{F}_4(\mathrm{x}) + \frac{1}{2}\log(3)\widehat{F}_2(\mathrm{x})$$

AdaBoost Example - Iteration 2

2.4 Update the weights to examples by using (1) again. For example, this time Case C has been correctly classified, so the new weight is

$$w_3^{new} = \frac{1}{2(1-\epsilon)} w_3^{old} = \frac{1}{2(1-2/8)} \frac{4}{8} = \frac{4}{12}$$

The updated weights are

Data	Weight Iter 1	Weight Iter 2	Weight Iter 3
A	1/5	1/8	1/12
B	1/5	1/8	1/12
C	1/5	1/2	4/12
D	1/5	1/8	3/12
E	1/5	1/8	3/12

2.5 Check whether the stopping criteria are met. If yes, stop; otherwise go to step 2.5

AdaBoost Example - Iteration 3

3.1 Use the new weights from Loop 2.

3.2 Again, calculate misclassification rate ϵ_k for each classifier $\widehat{F}_k(x)$.

3.3 Pick $\widehat{F}_k(x)$ with the lowest misclassification rate. In this case, it is $\widehat{F}_6(x)$.

Data	Weight Iter 1	Weight Iter 2	Weight Iter 3
A	1/5	1/8	1/12
B	1/5	1/8	1/12
C	1/5	1/2	4/12
D	1/5	1/8	3/12
E	1/5	1/8	3/12

Classifiers	Misclassified	Rate
$x < 1$	A, B, D, E	1/12+1/12+1/4+1/4 = 2/3
$x < 2$	D, E	1/4+1/4 = 1/2
$x < 6$	C, D, E	1/3+1/4+1/4 = 5/6
$x \geq 1$	C	1/3 = 1/3
$x \geq 2$	A, B, C	1/12+1/12+1/3 = 1/2
$x \geq 6$	A, B	1/12+1/12 = 1/6

AdaBoost Example - Iteration 3

3.4 Calculate voting power for the best classifier $\widehat{F}_6(x)$.

$$\alpha_k = \frac{1}{2}\log\left(\frac{1-\epsilon_k}{\epsilon_k}\right) = \frac{1}{2}\log(5)$$

and construct the current best ensemble

$$\frac{1}{2}\log(4)\widehat{F}_4(x) + \frac{1}{2}\log(3)\widehat{F}_2(x) + \frac{1}{2}\log(5)\widehat{F}_6(x)$$

AdaBoost Example - Stopping

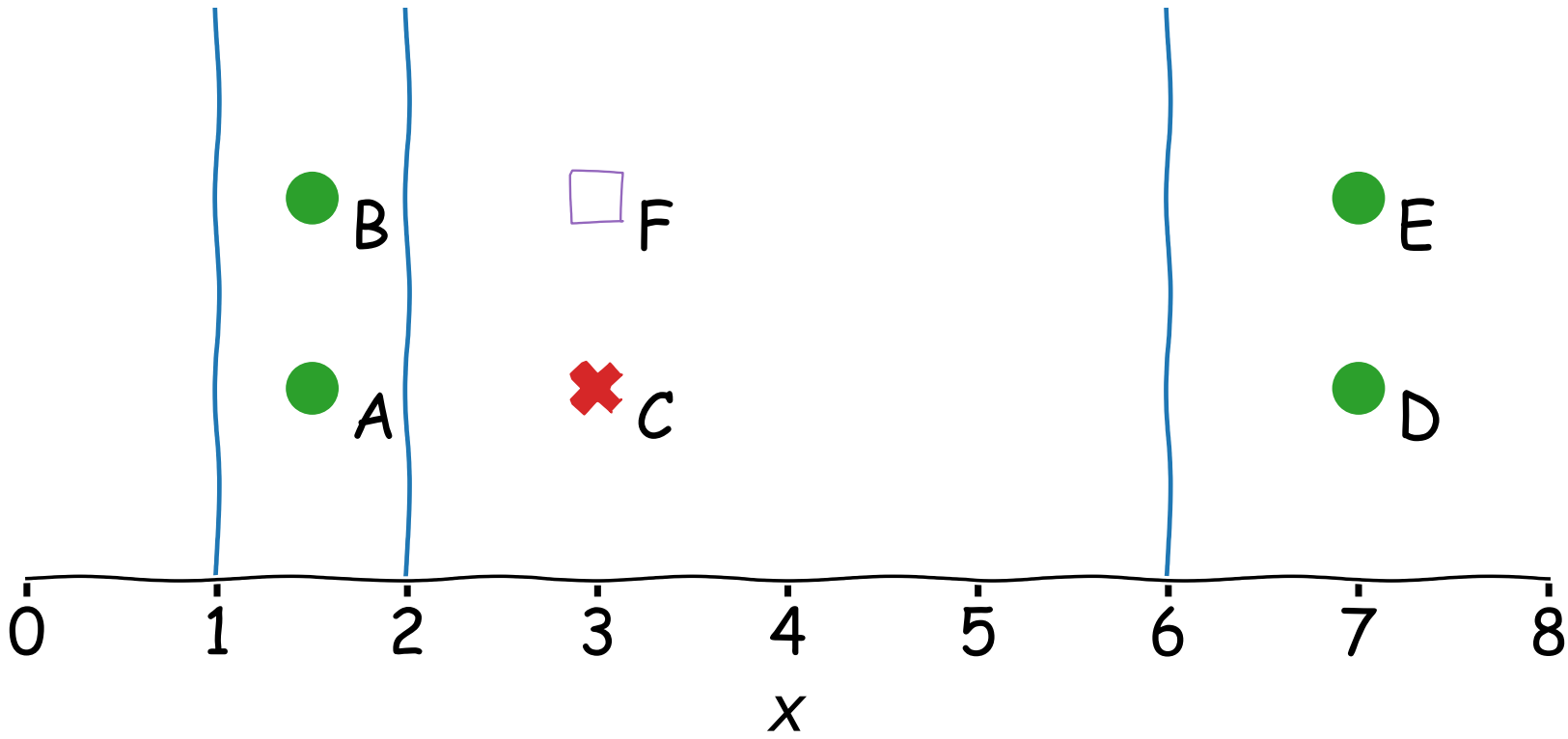
We stop here since our current classifier

$$\overline{F}(x) = \text{sgn}\left(\frac{1}{2}\log(4)\widehat{F}_4(x) + \frac{1}{2}\log(3)\widehat{F}_2(x) + \frac{1}{2}\log(5)\widehat{F}_6(x)\right)$$

can classify all training examples correctly.

AdaBoost Example - Prediction

Use the final classifier to predict the new case $F = 3$ shown below



AdaBoost Example - Prediction

Classifiers	F_k(3)
$x < 2$	-1
$x \geq 1$	1
$x \geq 6$	-1

$$\overline{F}(x) = \text{sgn}\left(\frac{1}{2}\log(4)\widehat{F}_4(3) + \frac{1}{2}\log(3)\widehat{F}_2(3) + \frac{1}{2}\log(5)\widehat{F}_6(3)\right)$$

$$\overline{F}(x) = \text{sgn}\left(\frac{1}{2}\log(4)(1) + \frac{1}{2}\log(3)(-1) + \frac{1}{2}\log(5)(-1)\right)$$

$$\overline{F}(x) = \text{sgn}\left(\frac{1}{2}\log(4) - \frac{1}{2}\log(3) - \frac{1}{2}\log(5)\right)$$

$$\overline{F}(x) = \text{sgn}\left(-0.6609\right)$$

$$\overline{F}(x) = -1$$

AdaBoost

Discrete AdaBoost

It is more common to use "Discrete AdaBoost".

1. Initialise weights $w_i = \frac{1}{N}$ for all data points i
2. Repeat M times:
   - 2.1 Fit a classifier $f(x)$ to training data using weights w_i
   - 2.2 Compute the normalized weighted misclassification rate

$$\epsilon_m = \frac{\sum_i^N w_i (f_k(x_i) \neq y_i)}{\sum_i^N w_i}$$

   - 2.3 Update α_m

$$\alpha_m = \log\left(\frac{1-\epsilon_m}{\epsilon_m}\right)$$

   - 2.4 Update every w_i

$$w_i = w_i \exp(\alpha_m \cdot (f_k(x_i) \neq y_i))$$

Discrete AdaBoost

Predictions are given by

$$\overline{F}(x) = \text{sgn}\left(\sum_{m=1}^{M} \alpha_m f_m(x)\right)$$

Generalising AdaBoost

AdaBoost and other boosting algorithms are just fitting a weighted sum of models i.e.

$$\overline{F}(x) = \sum_{m=1}^{M} \alpha_m f_m(x)$$

It is not hard to imagine that in the regression case a complicated polynomial could be accurately represented by the sum of many simpler functions.

Generalising AdaBoost

It has been shown that AdaBoost selects $f_m(x)$ and α_m under the **exponential loss** function.

For an accessible derivation I recommend reading Babenko, B, *Note: A Derivation of Discrete AdaBoost*.

A copy has been provided on Canvas.

Real AdaBoost

1. Initialise weights $w_i = \frac{1}{N}$ for all data points i
2. Repeat M times:
   - 2.1 Fit a classifier that generates $p_m(y = 1|x) = p_m(x)$ to training data using weights w_i
   - 2.2 Set $f_m(x) = \frac{1}{2} \log \frac{p_m(x)}{1-p_m(x)}$
   - 2.3 Update every w_i

$$w_i = w_i \exp(-y_i f_m(x))$$

   - 2.4 Normalize w_i

$$w_i = \frac{w_i}{\sum_i^N w_i}$$

Real AdaBoost

In practice Real AdaBoost tends to outperform Discrete AdaBoost

Predictions are given by

$$\overline{F}(x) = \text{sgn}\left(\sum_{m=1}^{M} f_m(x)\right)$$

Wrapping Up

AdaBoost Context

AdaBoost (Adaptive Boosting) was developed in the mid 90s, amidst a wave of boosting algorithm research.

The success of AdaBoost was so profound that the originators were awarded the Godel Prize, which is a prize for contributions to theoretical computer science.

AdaBoost inspired many developments in the field of boosting, such as Gradient Boosting.

AdaBoost Advantages

- plug and play with any classifier
- classifiers can be entirely different models
- classifiers need only be better than random guessing
- simple and fast training procedure

AdaBoost Disadvantages

- it can be sensitive to noise and outliers
- potential for overfitting
- since it is trained sequentially the training procedure can't be performed in parallel
- loss of interpretability compared to single models