Queen Mary
University of London

ADRIAN BEVAN

ATLAS UK MEETING 2020: MACHINE LEARNING SESSION

DECISION TREES

LECTURE PLAN

- Introduction
- Decision Trees
  - Boosting
- Suggested tools
- Suggested reading

Aim: Provide sufficient minimal background for you to (i) understand what a boosted decision tree is and (ii) train & optimise a boosted decision tree.

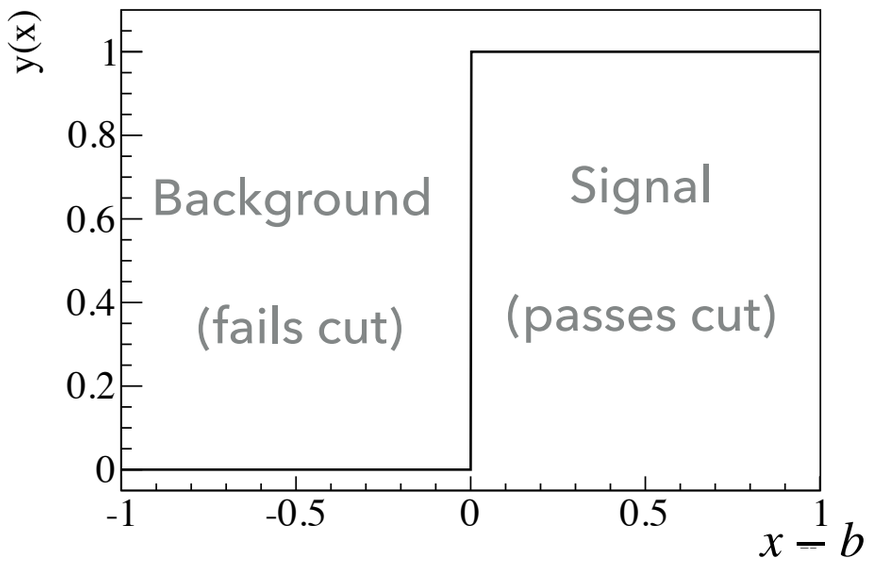
Focus on the use of the AdaBoost.M1 algorithm.

INTRODUCTION

- The cut based and linear discriminant analysis methods are limited.
- The underlying concepts of applying Heavyside function constraints on data selection and on the use of a decision boundary definition (a plane in hyperspace) of the form of the dot product $\alpha^T x + \beta$ can be applied in more complicated algorithms.
- Here we consider extension to the concept of rectangular cuts to decision trees as a machine learning algorithm.
  - We will have to introduce the concepts of classification and regression; and methods to mitigate mis-classification of data.
  - The issue of overtraining is something we will come back to when discussing optimisation.

DECISION TREES

- Consider a data sample with an N dimensional input feature space X.
- X can be populated by examples from two or more different species of event (also called classes, categories or types).
- Consider the two types and call them signal and background, respectively*.
- We can use a Heavyside function to divide the data into two parts:
- We can use this to distinguish between regions populated signal and background:



For an arbitrary cut position in x we can modify the Heavyside function

$$H(x) = \frac{1}{2}(1 + sign(x - b))$$

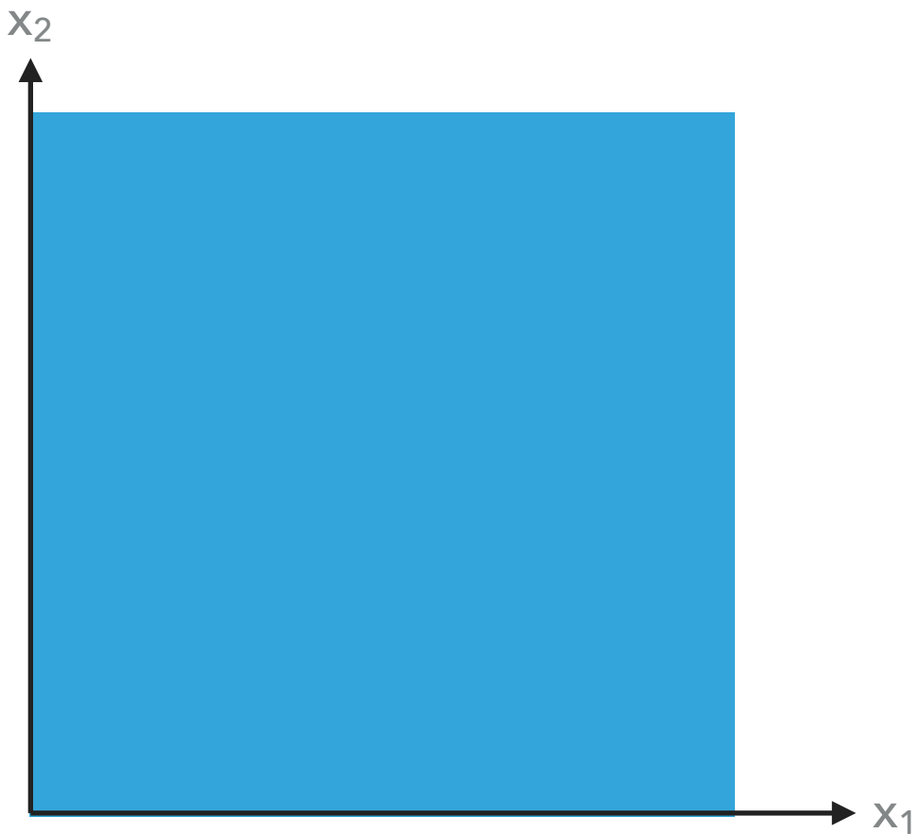
where b is the offset (bias) from zero.

*Can generalise the problem to an arbitrary number of types.

DECISION TREES

- Decision trees divide the data feature space into a set of hypercubes that are classified as signal (+1) or background (-1) like.
  - Each region can be fitted with a constant to represent the data in that region.
  - We can recursively continue to sub-divide the data until some minimum number of examples are left in each sub-division.

Can describe the data as the root node.

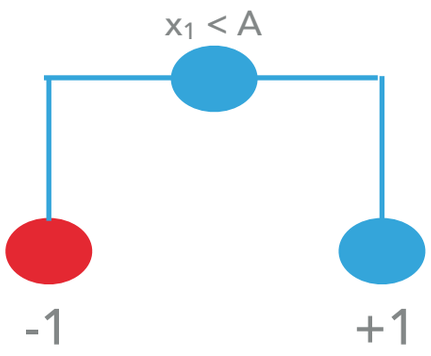


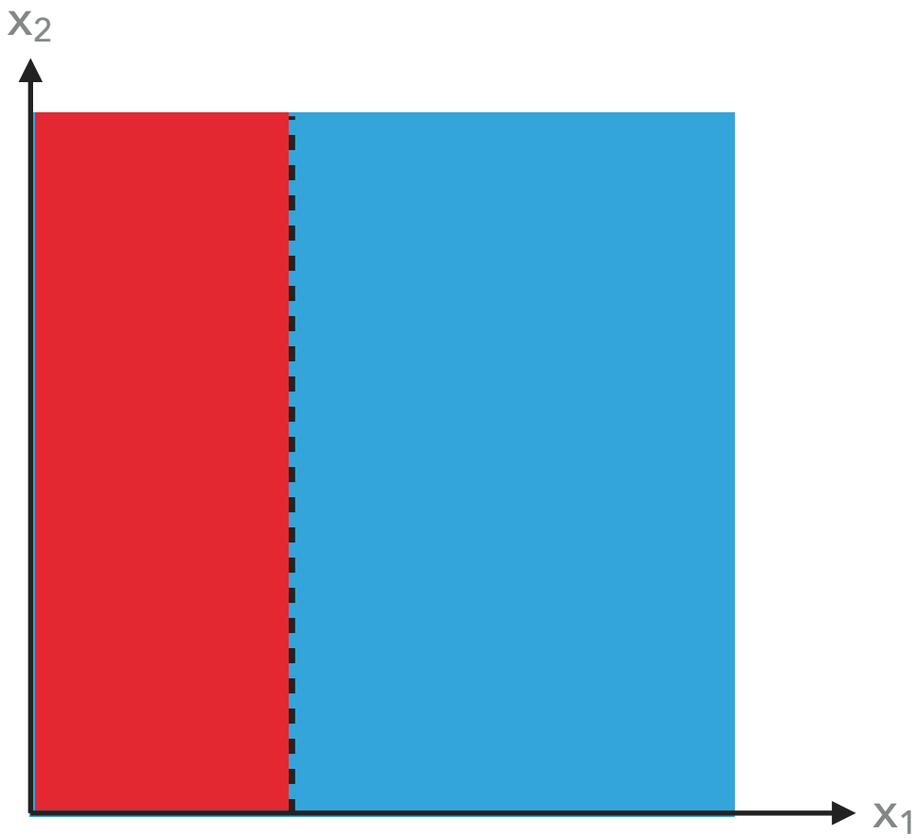
Example feature space described by $X=\{x_1, x_2\}$

DECISION TREES

- Decision trees divide the data feature space into a set of hypercubes that are classified as signal (+1) or background (-1) like.
  - Each region can be fitted with a constant to represent the data in that region.
  - We can recursively continue to sub-divide the data until some minimum number of examples are left in each sub-division.

The data get divided into two partitions.



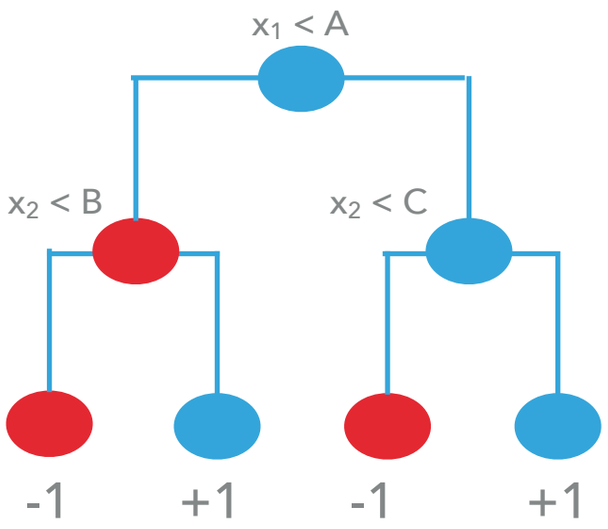


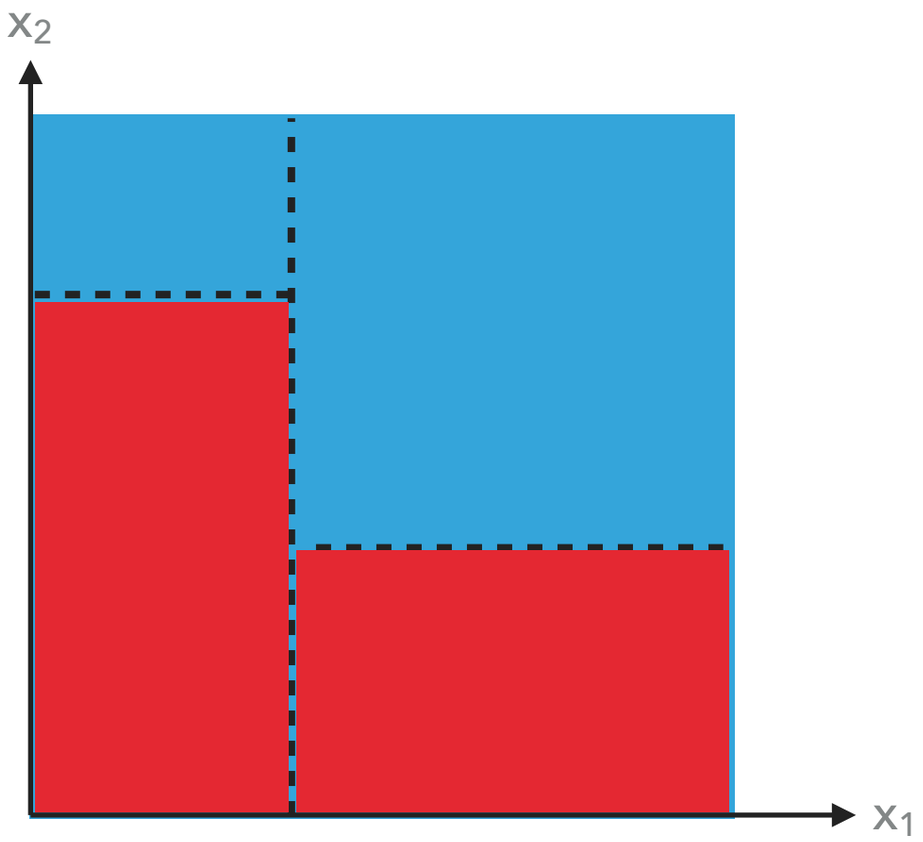
Cut on the feature space to separate the data into two different regions.

DECISION TREES

- Decision trees divide the data feature space into a set of hypercubes that are classified as signal (+1) or background (-1) like.
  - Each region can be fitted with a constant to represent the data in that region.
  - We can recursively continue to sub-divide the data until some minimum number of examples are left in each sub-division.

Divide the data again



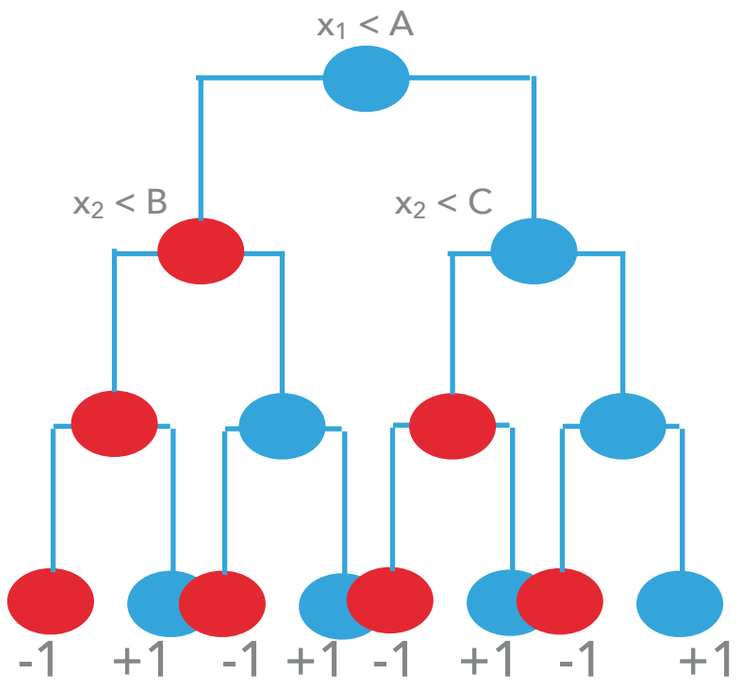


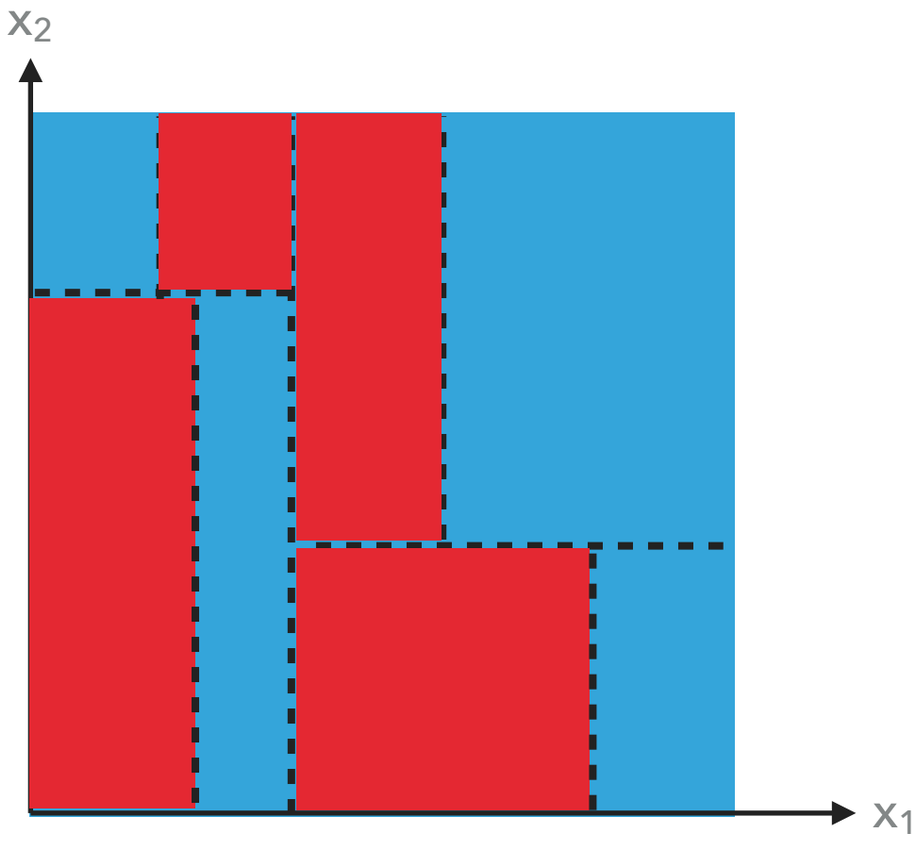
The feature space gets further sub-divided.

DECISION TREES

- Decision trees divide the data feature space into a set of hypercubes that are classified as signal (+1) or background (-1) like.
  - Each region can be fitted with a constant to represent the data in that region.
  - We can recursively continue to sub-divide the data until some minimum number of examples are left in each sub-division.

Divide the data again

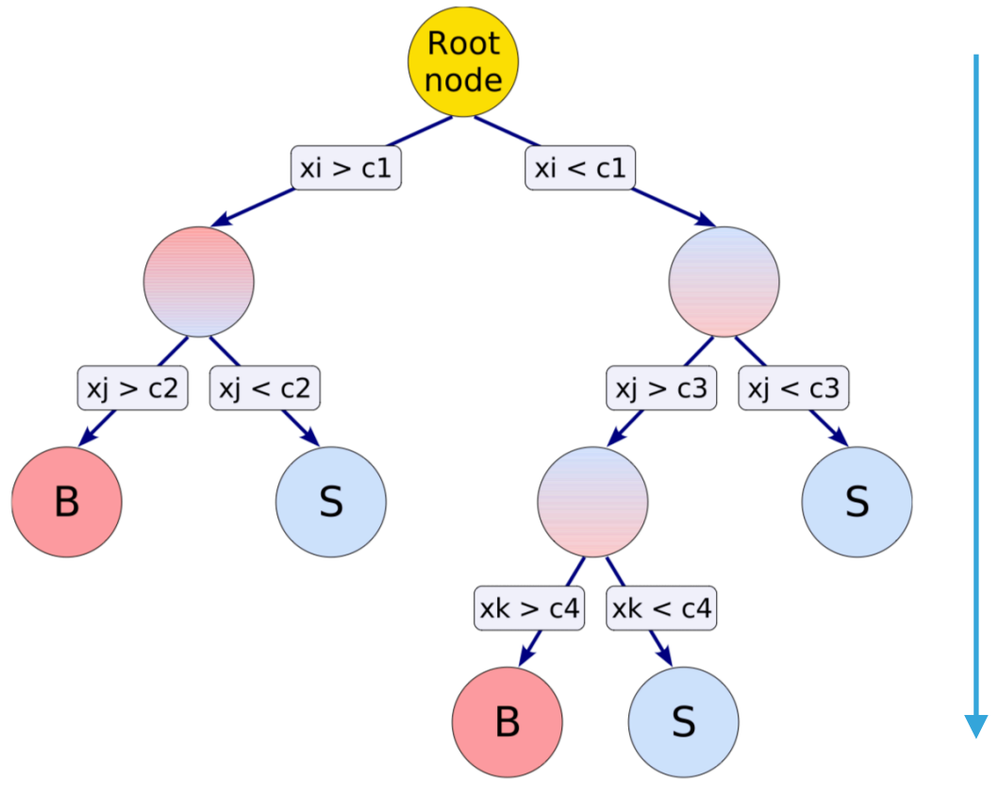




The feature space gets further sub-divided (again).

DECISION TREES

- The set of rectangular cuts applied to the data allow us to build a tree from the root note.
- We can impose limits on:
  - Tree depth (how many divisions are performed).
  - Node size (how many examples per partition).
- Trees can be extended to more than 2 categories.
- They lend themselves to classifying examples or adapted to make a quantitative prediction (regression)

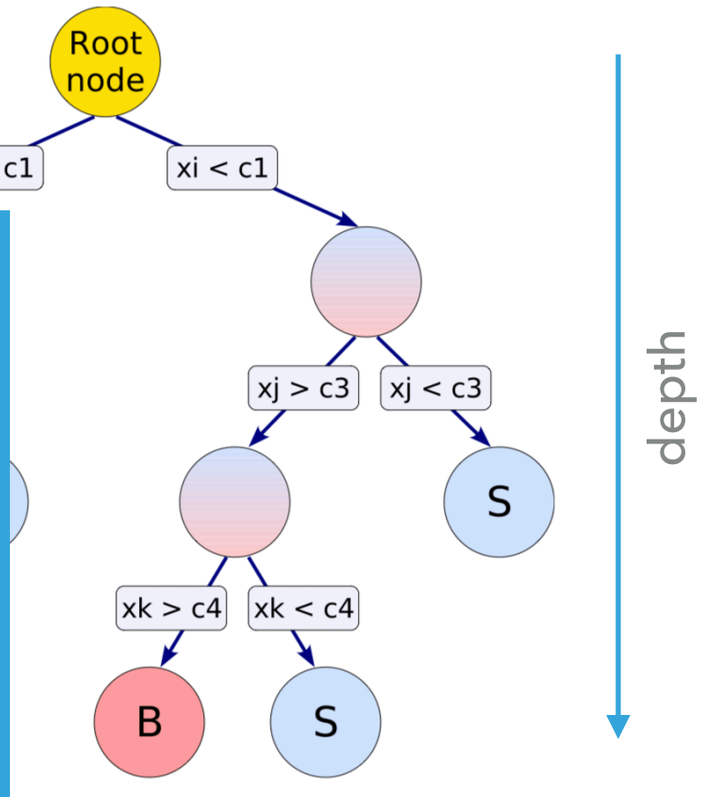


The decision tree output for a classification problem is

$$G(x) = +1 \text{ or } -1$$

DECISION TREES

- The set of rectangular cuts applied to the data allow us to build a tree from the root note.
- We can impose limit
  - Tree depth (how
    performed).
  - Node size (how m
    partition).
- Trees can be extend
  categories.
- They lend themselves to classifying examples or adapted to make a quantitative prediction (regression)



A Decision tree is what is known as a weak learner. It can take the input features in X, even when they are weakly separating, and combine those features to increase the separation.

A single tree is susceptible to overtraining (learning the statistical fluctuations in the training data)

As we shall see weak learners can be combined

The decision tree output for a classification problem is

$$G(x) = +1 \text{ or } -1$$

BOOSTING

- If a training example has been mis-classified in a training epoch, then the weight of that event can be increased for the next training epoch; so that the cost of mis-classification increases.
- The underlying aim is to take a weak learner and try and boost this into becoming a strong learner.
  - This example re-weighting technique is called boosting.
  - There are several re-weighting methods commonly used; here we discuss:
    - AdaBoost.M1 (popular variant of the Adaptive boosting method)
- Boosted Decision Trees are known as BDTs

BOOSTING: AdaBoost.M1

- i is the i^{th} example out of a data set with N examples.
- m is the m^{th} training out of an ensemble of M learners to be trained.

- Step 1:
  - Assign event weights of $w_i = 1/N$ to all of the N examples.

Freund and Schapire J. Jap. Soc. AI **14** (1999) 771-780

BOOSTING: AdaBoost.M1

- i is the i^{th} example out of a data set with N examples.
- m is the m^{th} training out of an ensemble of M learners to be trained.

- Step 1:
  - Assign event weights of $w_i = 1/N$ to all of the N examples.

- Step 2: for m=1 through M
  - Train the weak learner (in our case this is a BDT): $G_m(x)$.
  - Compute the error rate ε_m.
  - Calculate the boost factor $\beta_m = \frac{\varepsilon_m}{1-\varepsilon_m}$.
  - Update weights for misclassified examples $w_i \mapsto w_i e^{\ln(1/\beta_m)}$.

Freund and Schapire J. Jap. Soc. AI **14** (1999) 771-780

BOOSTING: AdaBoost.M1

- i is the i^{th} example out of a data set with N examples.
- m is the m^{th} training out of an ensemble of M learners to be trained.

- Step 1:
  - Assign event weights of $w_i = 1/N$ to all of the N examples.

- Step 2: for m=1 through M
  - Train the weak learner (in our case this is a BDT): $G_m(x)$.
  - Compute the error rate ε_m.
  - Calculate the boost factor $\beta_m = \frac{\varepsilon_m}{1-\varepsilon_m}$.
  - Update weights for misclassified examples $w_i \mapsto w_i e^{\ln(1/\beta_m)}$.

- Step 3:
  - Return the weighted committee: a combination of the M trees that have been learned from the data:

$$G(x) = sign\left(\sum_{m=1}^{M} ln\left[\frac{1-\varepsilon_m}{\varepsilon_m}\right] G_m(x)\right)$$

Freund and Schapire J. Jap. Soc. AI **14** (1999) 771-780

BOOSTING: AdaBoost.M1

- m=1 **INITIAL TRAINING SAMPLE** → $G_1(x)$
- m=2 **WEIGHTED SAMPLE** → $G_2(x)$
- m=3 **WEIGHTED SAMPLE** → $G_3(x)$

The $G_m(x)$ are individual weak learners; each is derived from a training using the data.

The m=1 training uses the original data; all subsequent trainings use reweighted data.

- m=M **WEIGHTED SAMPLE** → $G_M(x)$

$$G(x) = sign\left(\sum_{m=1}^{M} ln\left[\frac{1-\varepsilon_m}{\varepsilon_m}\right] G_m(x)\right)$$

The final classifier output is formed from a committee that is a weighted majority vote algorithm.

Freund and Schapire J. Jap. Soc. AI **14** (1999) 771-780

SUGGESTED TOOLS

- Many decision tree tools exist that you may wish to explore. A selection of these are linked below:
  - TMVA ← We will use this one today
  - SciKitLearn
  - XGBoost
- In addition to Python and C++ based tools, you will find decision trees implemented in R, Matlab, Mathmatica etc.

SUGGESTED READING (NON-HEP)

- In addition to the references given in the slides you may be interested in:
  - Hastie, Tibshirani and Friedman, The Elements of Statistical Learning, Springer (2011).