Advanced Machine Learning

Lecture 4: Combining Models

Gwenn Englebienne
Mannes Poel

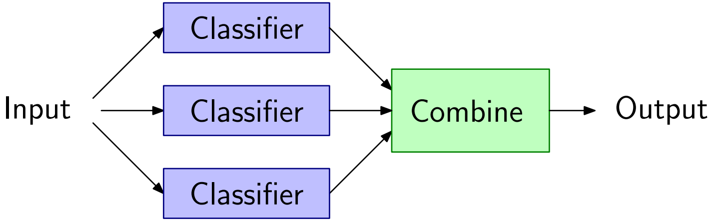
University of Twente

Combining models

What are committees?

- Traditional approach: train a classifier to predict a class
- Committee: Combine the output of multiple classifiers
  - For example, average the outputs
  - Alternatively, create a "meta-classifier"

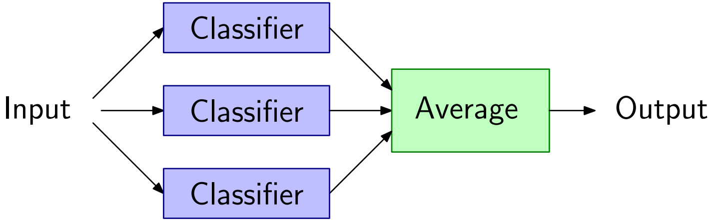
Combining models



What are committees?

- Traditional approach: train a classifier to predict a class
- Committee: Combine the output of multiple classifiers
  - For example, average the outputs
  - Alternatively, create a "meta-classifier"

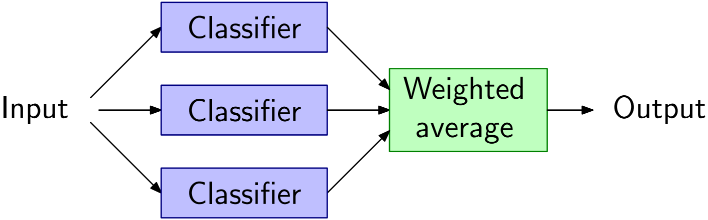
Combining models



What are committees?

- Traditional approach: train a classifier to predict a class
- Committee: Combine the output of multiple classifiers
  - For example, average the outputs
  - Alternatively, create a "meta-classifier"

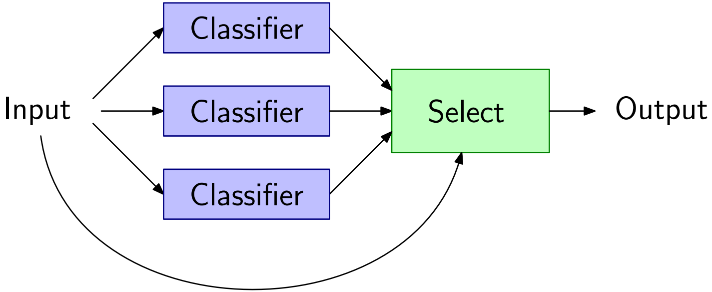
Combining models



What are committees?

- Traditional approach: train a classifier to predict a class
- Committee: Combine the output of multiple classifiers
  - For example, average the outputs
  - Alternatively, create a “meta-classifier”

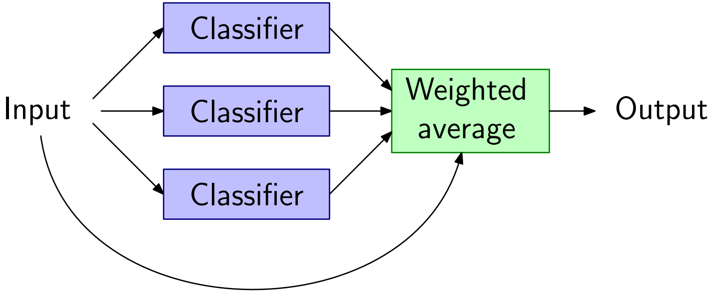
Combining models



What are committees?

- Traditional approach: train a classifier to predict a class
- Committee: Combine the output of multiple classifiers
  - For example, average the outputs
  - Alternatively, create a "meta-classifier"

Combining models



What are committees?

- Traditional approach: train a classifier to predict a class
- Committee: Combine the output of multiple classifiers
  - For example, average the outputs
  - Alternatively, create a "meta-classifier"

Why committees?

Consider M regression models $y_m(\mathbf{x}), 1 \leq m \leq M$ predicting $h(\mathbf{x})$. Each individual prediction error is

$$\epsilon_m(\mathbf{x}) = h(\mathbf{x}) - y_m(\mathbf{x}) \ ,$$

with an averaging committee:

$$y(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} y_m(\mathbf{x})$$

Why committees?

The expected sum-squared errors are:

Individual model (average)	Committee
$E_{AV} = \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\mathbf{x}}[\epsilon_m^2(\mathbf{x})]$	$E_{COM} = \mathbb{E}_{\mathbf{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}\epsilon_m(\mathbf{x})\right)^2\right]$

so that, if the errors are uncorrelated, we get

$$E_{COM} = \frac{1}{M^2}\mathbb{E}_{\mathbf{x}}\Big[\sum_{m=1}^{M}\epsilon_m^2(\mathbf{x}) + 2\sum_{m\neq n}\epsilon_m(\mathbf{x})\epsilon_n(\mathbf{x})\Big] \qquad = \frac{1}{M}E_{AV}$$

Why committees?

The expected sum-squared errors are:

Individual model (average)	Committee
$E_{AV} = \frac{1}{M}\sum_{m=1}^{M}\mathbb{E}_{\mathbf{x}}[\epsilon_m^2(\mathbf{x})]$	$E_{COM} = \mathbb{E}_{\mathbf{x}}\left[\left(\frac{1}{M}\sum_{m=1}^{M}\epsilon_m(\mathbf{x})\right)^2\right]$

so that, if the errors are uncorrelated, we get

$$E_{COM} = \frac{1}{M^2}\mathbb{E}_{\mathbf{x}}\Big[\sum_{m=1}^{M}\epsilon_m^2(\mathbf{x}) \underbrace{+2\sum_{m\neq n}\epsilon_m(\mathbf{x})\epsilon_n(\mathbf{x})\Big]}_{=0} \qquad = \frac{1}{M}E_{AV}$$

Why committees?

In theory, committees can vastly reduce the expected error of individual classifiers

- Make the expected error arbitrarily small by increasing M
- In practice, the classifiers are highly correlated
  - The error reduction is then small
- But: it can be shown that

$$E_{AV} \geq E_{COM}$$

- We can improve the performance of committees by decreasing the correlation between the classifiers

Bias-Variance decomposition

Consider multiple training data sets $D = \{(\mathbf{x}_n, h(\mathbf{x}_n))\}$ of fixed size
Taking the expected squared loss of a model, we can decompose:

$$\mathbb{E}_D[(y_D(\mathbf{x}) - \hat{t}(\mathbf{x}))^2] = \underbrace{(\mathbb{E}_D[y_D(\mathbf{x})] - \hat{t}(\mathbf{x}))^2}_{\text{bias}^2} + \underbrace{\mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})])^2]}_{\text{variance}}$$

Interpretation:

- The bias captures how well the model *can* perform.
  Flexible models will have low bias.
- The variance captures how much the end model depends on the specific dataset.
  Flexible models will have high variance.

Bias-Variance decomposition

Consider multiple training data sets $D = \{(\mathbf{x}_n, h(\mathbf{x}_n))\}$ of fixed size
Taking the expected squared loss of a model, we can decompose:

$$\mathbb{E}_D[(y_D(\mathbf{x}) - \hat{t}(\mathbf{x}))^2] = \underbrace{(\mathbb{E}_D[y_D(\mathbf{x})] - \hat{t}(\mathbf{x}))^2}_{\text{bias}^2} + \underbrace{\mathbb{E}_D[(y_D(\mathbf{x}) - \mathbb{E}_D[y_D(\mathbf{x})])^2]}_{\text{variance}}$$

Interpretation:

- The bias captures how well the model *can* perform.
  Flexible models will have low bias.
- The variance captures how much the end model depends on the specific dataset.
  Flexible models will have high variance.

Bias-Variance decomposition

Bias-Variance decomposition:

- Gives us insight into how a particular model generalises
  - High bias-low variance models do not learn from the data
  - Low bias-high variance models overfit on the training data
  - Optimal model flexibility (e.g., regularisation): good bias–variance trade-off.
- Has little practical value: single training dataset
- Provides insight into why committees are useful

Optimal ensemble learning

For best ensemble performance, we want the base learners to be flexible enough and as diverse as possible

Bias-Variance decomposition

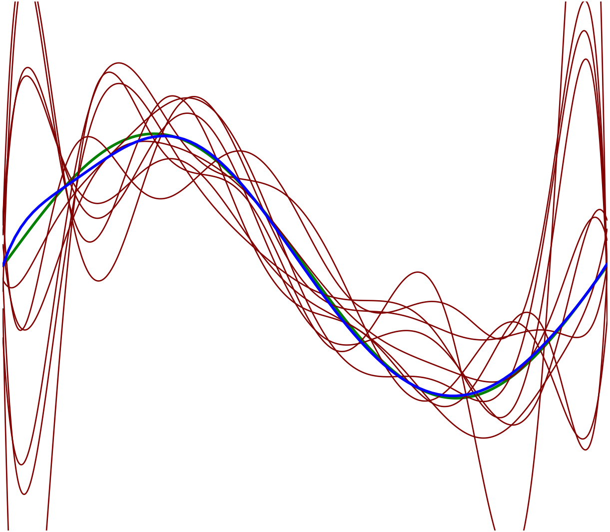
Bias-Variance decomposition:

- Gives us insight into how a particular model generalises
  - High bias-low variance models do not learn from the data
  - Low bias-high variance models overfit on the training data
  - Optimal model flexibility (e.g., regularisation): good bias–variance trade-off.
- Has little practical value: single training dataset
- Provides insight into why committees are useful

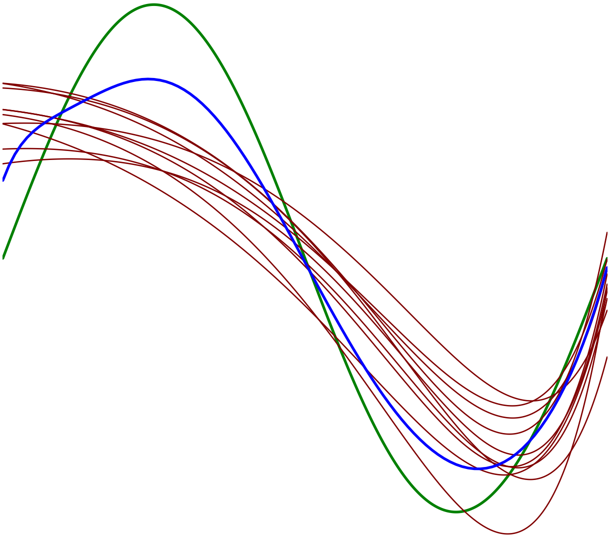
Optimal ensemble learning

For best ensemble performance, we want the base learners to be flexible enough and as diverse as possible

Bias-variance: an example



Bias-variance: an example



Making committees

Where do we get the base learners?

1. Single type of classifiers:
   Homogeneous learners
2. Multiple types of classifiers:
   Heterogeneous learners

Diversity in homogeneous learners?

- Subsample the training data
- Add randomness to the learning algorithm
- Manipulate attributes or outputs

Making committees

Where do we get the base learners?

1. Single type of classifiers:
   Homogeneous learners
2. Multiple types of classifiers:
   Heterogeneous learners

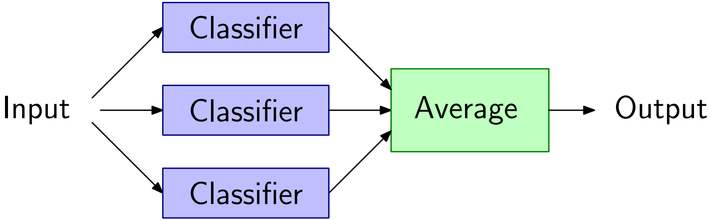
Diversity in homogeneous learners?

- Subsample the training data
- Add randomness to the learning algorithm
- Manipulate attributes or outputs

Bagging: Bootstrap Aggregation

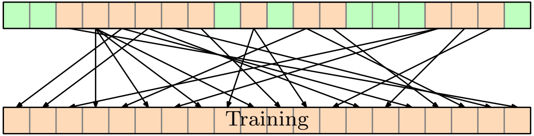
We rarely have infinitely large training datasets. . .

- Or infinitely many. . .
- Using bootstrapping, we can create new datasets
- The correlation between datasets is then known and kept small
- **Bootstrap aggregation**:
  Simply average the outcomes of classifiers trained on different bootstrap datasets



Bootstrapping

Sample N points at random from the data **with replacement**



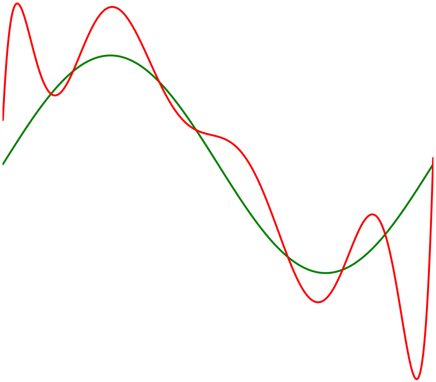
The probability for not picking a data point is

$$p(\neg b) = (1 - 1/N)^N \approx 0.368 \tag{1}$$

The expected number of used data points is therefore

$$p(b) = 1 - p(\neg b) \approx 0.632N \tag{2}$$

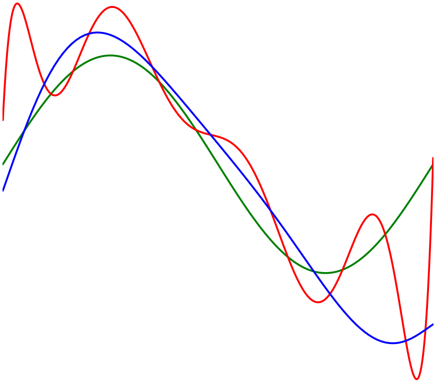
Bagging: an example



In this example:

- A polynomial was fitted to 10 noisy training points (red)
- 1000 polynomials were fitted to bootstrap sets from the same 10 datapoints and averaged (blue line)

Bagging: an example



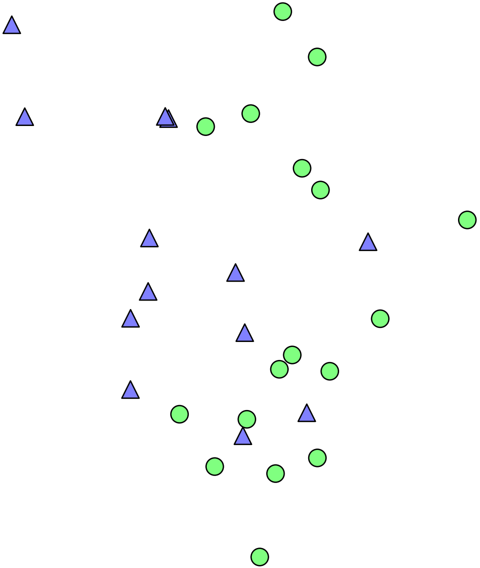
In this example:

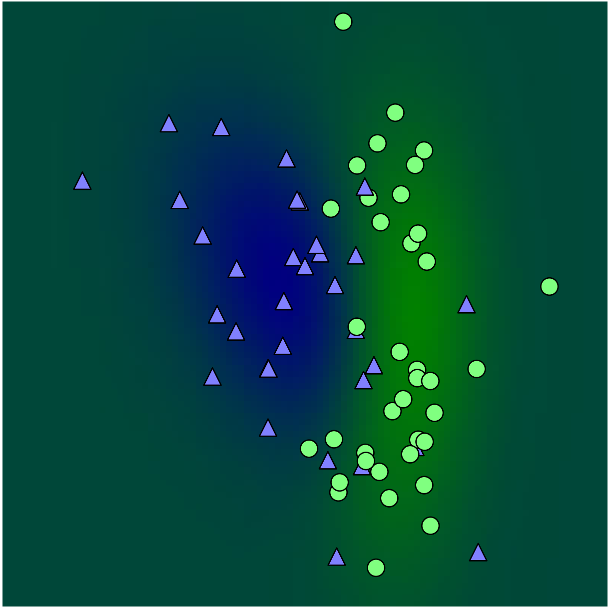
- A polynomial was fitted to 10 noisy training points (red)
- 1000 polynomials were fitted to bootstrap sets from the same 10 datapoints and averaged (blue line)

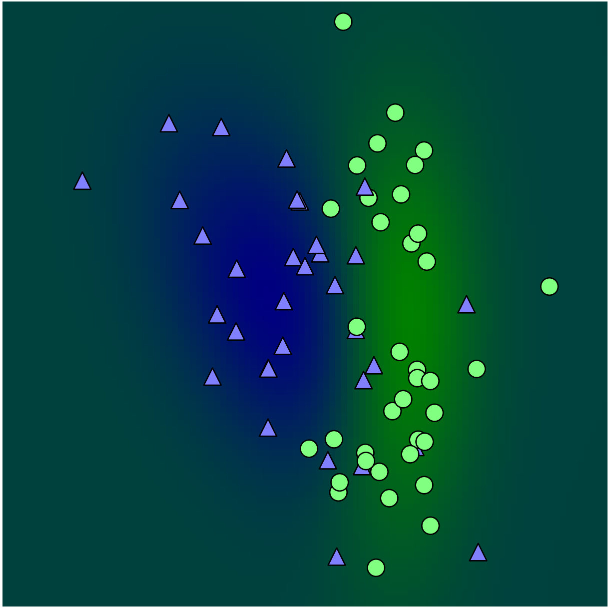
Bagging

Bagging

- Improves results with high-variance models
- No independent datasets ($\Rightarrow$ small improvements)
- Cannot help with high bias models







Boosting

Weak learner Learner that performs better than random

Strong learner Learner with accuracy $1 - \epsilon$, where ϵ is arbitrarily small

[Shapire 1990]: Weak learners in the same class as strong learners

Boosting

A technique to combine weak learners to form a strong learner

Boosting

Weak learner Learner that performs better than random

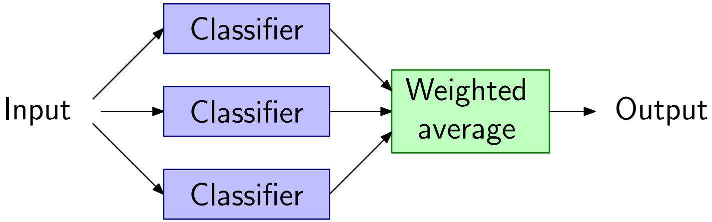
Strong learner Learner with accuracy $1 - \epsilon$, where ϵ is arbitrarily small

[Shapire 1990]: Weak learners in the same class as strong learners

Boosting

A technique to combine weak learners to form a strong learner

Adaptive boosting (Adaboost)



Adaptive boosting:

- Assign each training datapoint a weight
- Iterate:
  - Train a classifier based on the weighted training data
  - Assign this classifier a weight based on how well it performs
  - Update the datapoints' weights based on how many classifiers classify it correctly

Adaptive boosting: the algorithm

1. Set $w_n^{(1)} = \frac{1}{N}$
2. For $m = 1, \ldots, M$
   2.1 Fit $y_m(\mathbf{x})$ by minimising $E_m = \sum_{n \in \mathcal{M}_m} w_n^{(m)}$ (Total weight of missclassified points)
   2.2 Evaluate
   $$\epsilon_m = \frac{\sum_{n \in \mathcal{M}_m} w_n^{(m)}}{\sum_n w_n^{(m)}},$$
   (Ratio missclassified for that classifier)
   set $\alpha_m = \log \frac{1-\epsilon_m}{\epsilon_m}$
   2.3 Update the weights
   $$w_n^{(m+1)} = \begin{cases} w_n^{(m)} & \text{if } y_m(x_n) = t_n \\ w_n^{(m)} \exp \alpha_m & \text{Otherwise} \end{cases}$$
3. Classify the datapoints as
$$Y_M(\mathbf{x}) = \operatorname{sign} \sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})$$

Adaptive boosting: the algorithm

1. Set $w_n^{(1)} = \frac{1}{N}$
2. For $m = 1, \ldots, M$
   2.1 Fit $y_m(\mathbf{x})$ by minimising $E_m = \sum_{n \in \mathcal{M}_m} w_n^{(m)}$
   2.2 Evaluate

   $$\epsilon_m = \frac{\sum_{n \in \mathcal{M}_m} w_n^{(m)}}{\sum_n w_n^{(m)}},$$

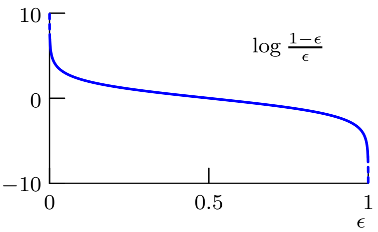
   set $\alpha_m = \log \frac{1-\epsilon_m}{\epsilon_m}$

   2.3 Update the weights

$$w_n^{(m+1)} = \begin{cases} w_n^{(m)} & \text{if } y_m(x_n) = t_n \\ w_n^{(m)} \exp \alpha_m & \text{Otherwise} \end{cases}$$

3. Classify the datapoints as

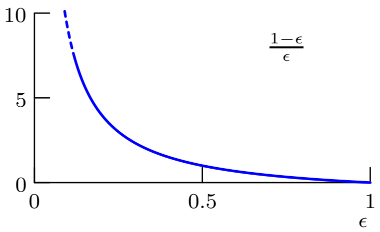
$$Y_M(\mathbf{x}) = \text{sign} \sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})$$



Adaptive boosting: the algorithm

UNIVERSITY OF TWENTE.
HMI.

1. Set $w_n^{(1)} = \frac{1}{N}$
2. For $m = 1, \ldots, M$
   2.1 Fit $y_m(\mathbf{x})$ by minimising $E_m = \sum_{n \in \mathcal{M}_m} w_n^{(m)}$
   2.2 Evaluate
   $$\epsilon_m = \frac{\sum_{n \in \mathcal{M}_m} w_n^{(m)}}{\sum_n w_n^{(m)}},$$
   set $\alpha_m = \log \frac{1-\epsilon_m}{\epsilon_m}$
   2.3 Update the weights



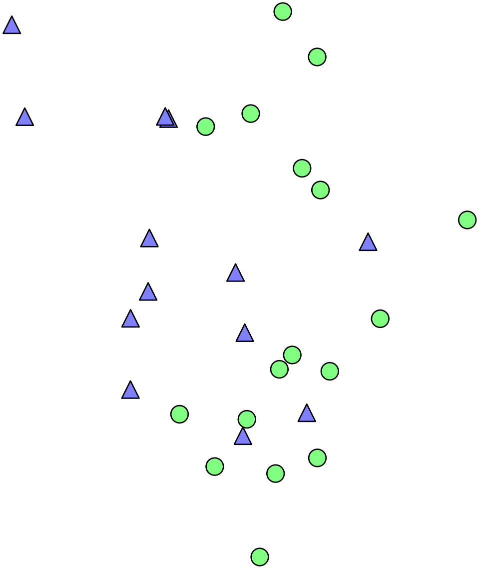
$$w_n^{(m+1)} = \begin{cases} w_n^{(m)} & \text{if } y_m(x_n) = t_n \\ w_n^{(m)} \exp \alpha_m & \text{Otherwise} \end{cases}$$

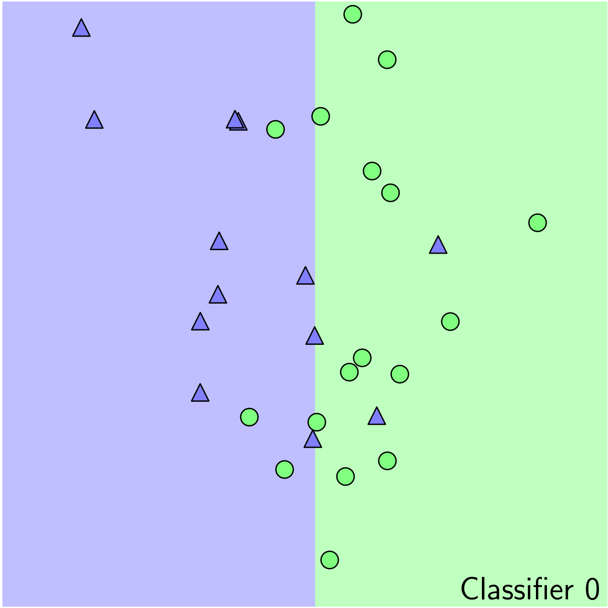
3. Classify the datapoints as

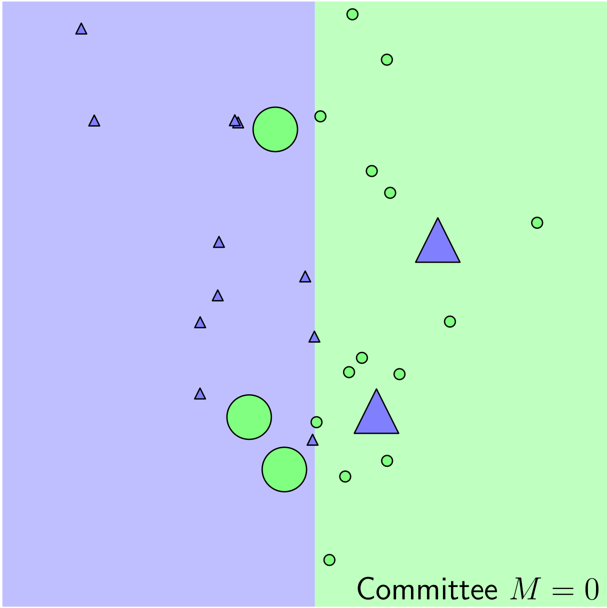
$$Y_M(\mathbf{x}) = \operatorname{sign} \sum_{m=1}^{M} \alpha_m y_m(\mathbf{x})$$

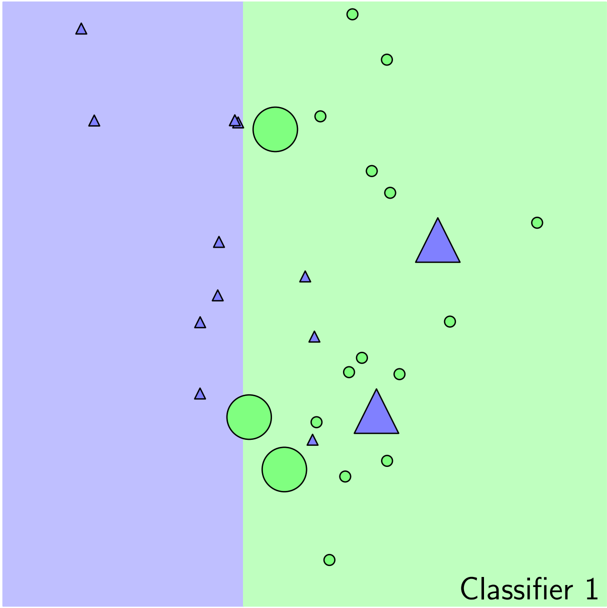
Adaboost example

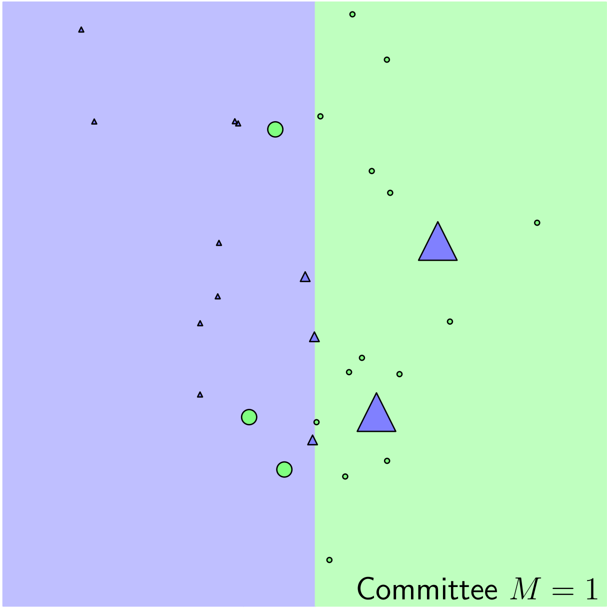
- Simple 2-class, 2-dimensional dataset
- Simple classifiers:
  - Choose threshold θ, dimension d, class c
  - $y(\mathbf{x}) = \begin{cases} \mathcal{C}_c & \text{iff } x_d < \theta \\ \mathcal{C}_{1-c} & \text{Otherwise} \end{cases}$
- Create ensemble using adaboost

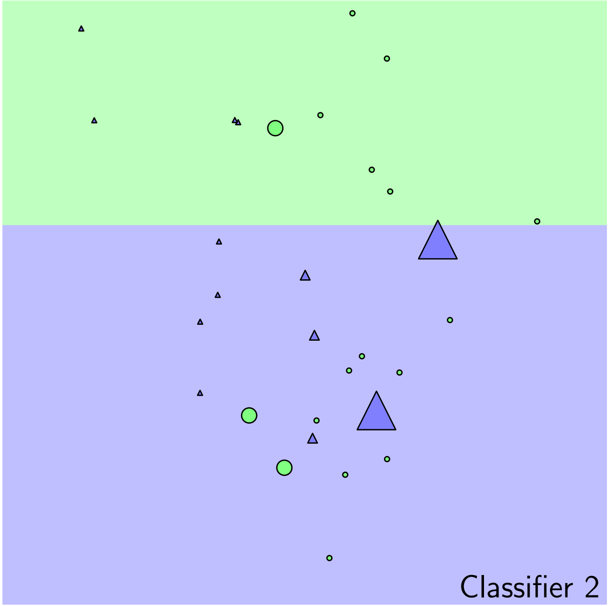


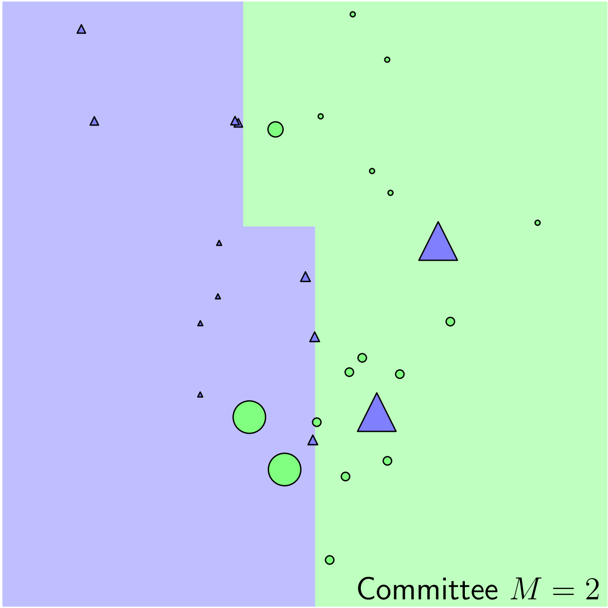

Classifier 0

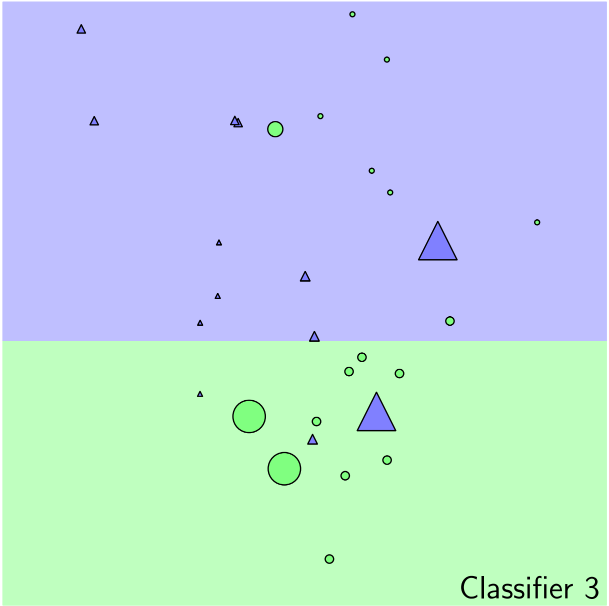

Committee $M = 0$

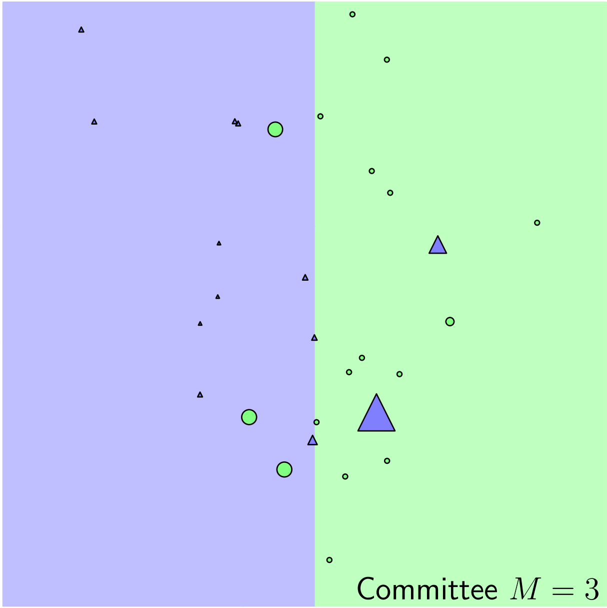

Classifier 1

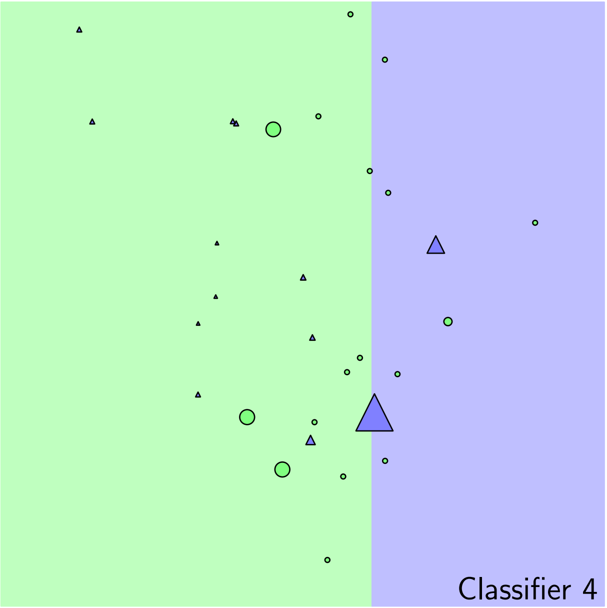

Committee $M = 1$

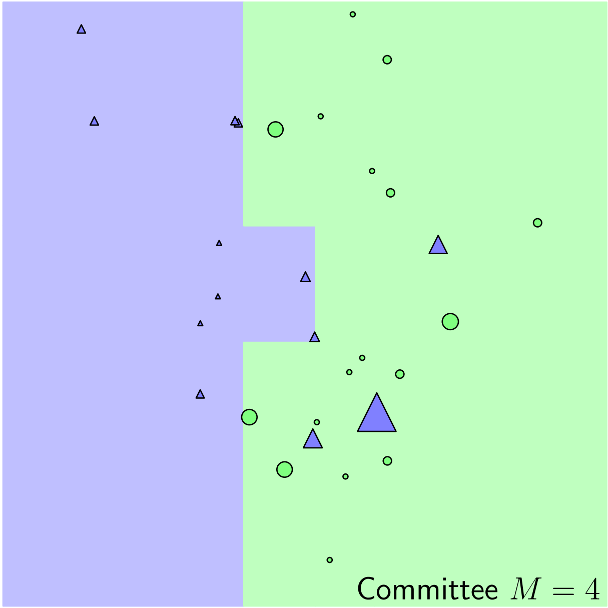

Classifier 2

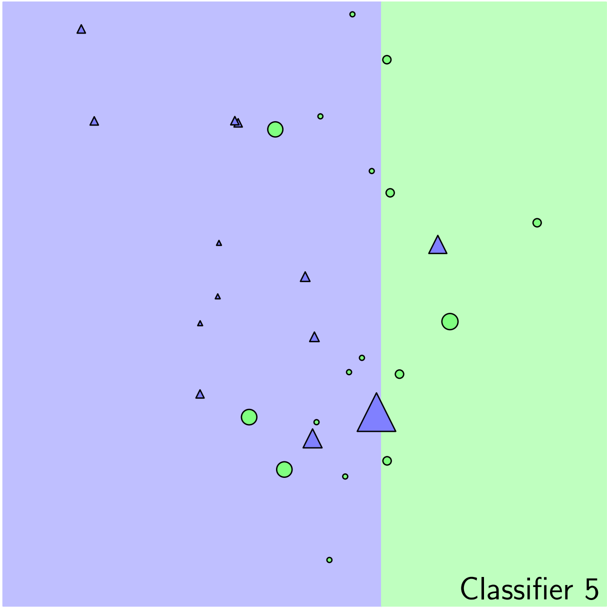

Committee $M = 2$

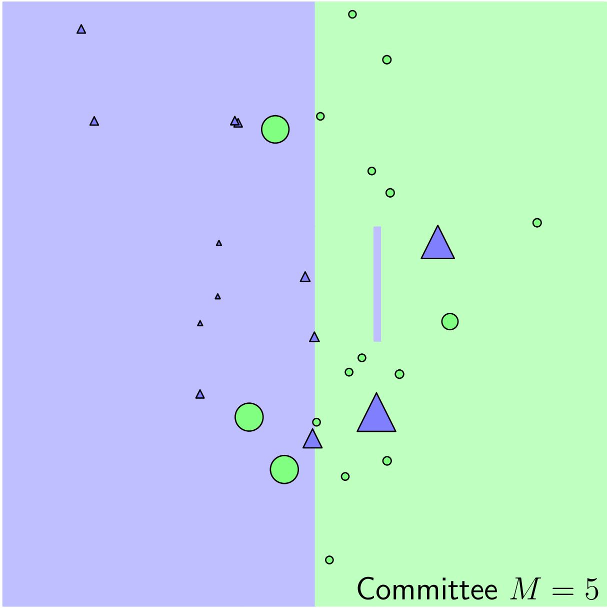

Classifier 3

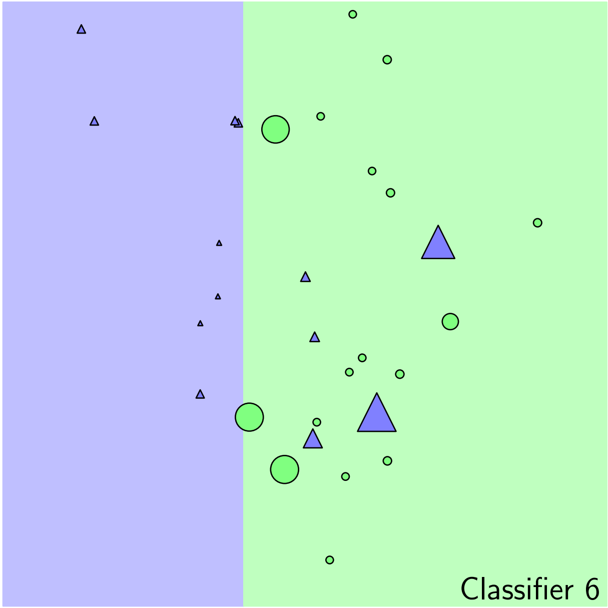

Committee $M = 3$

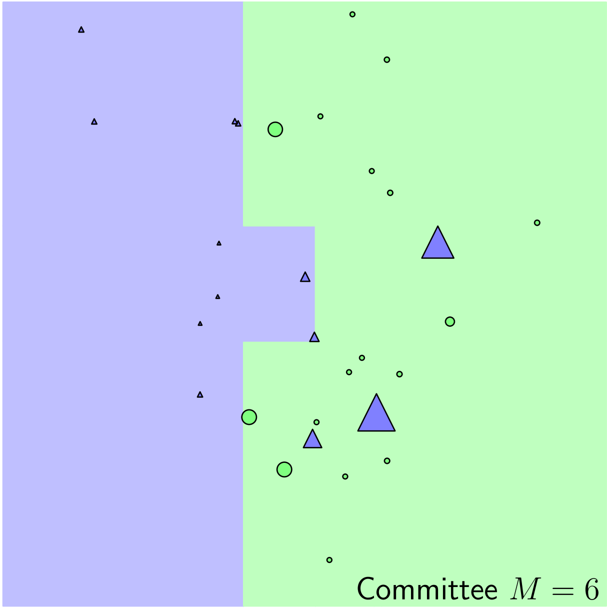

Classifier 4

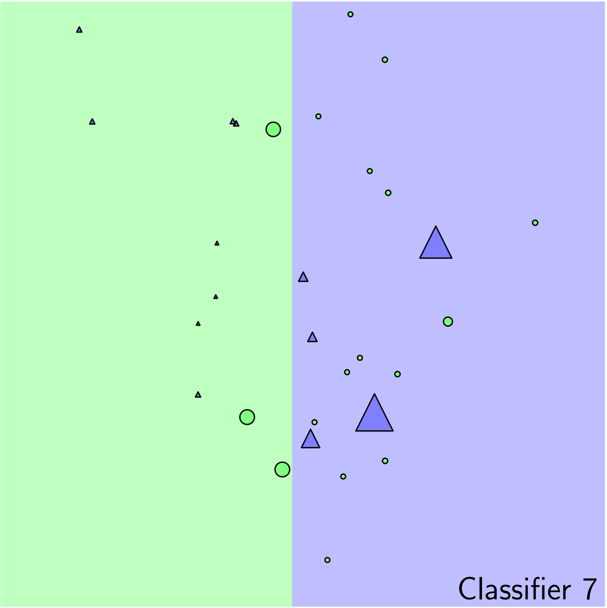

Committee $M = 4$

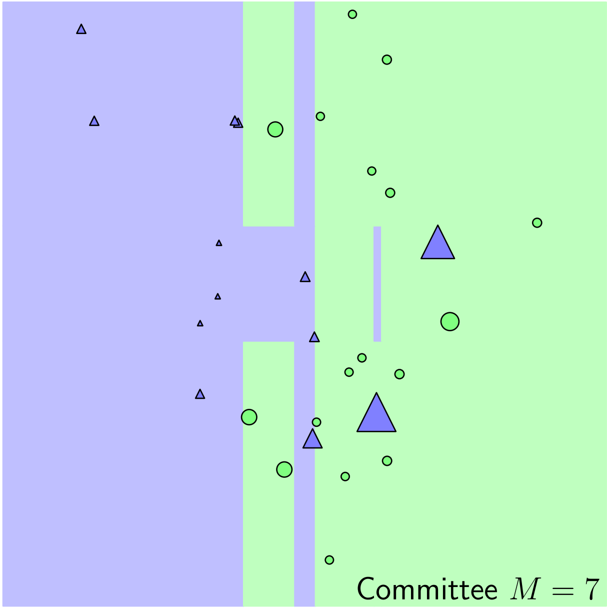

Classifier 5

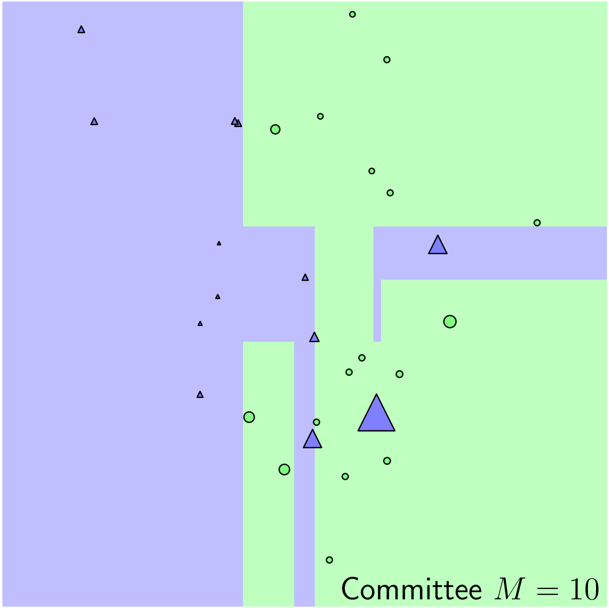

Committee $M = 5$

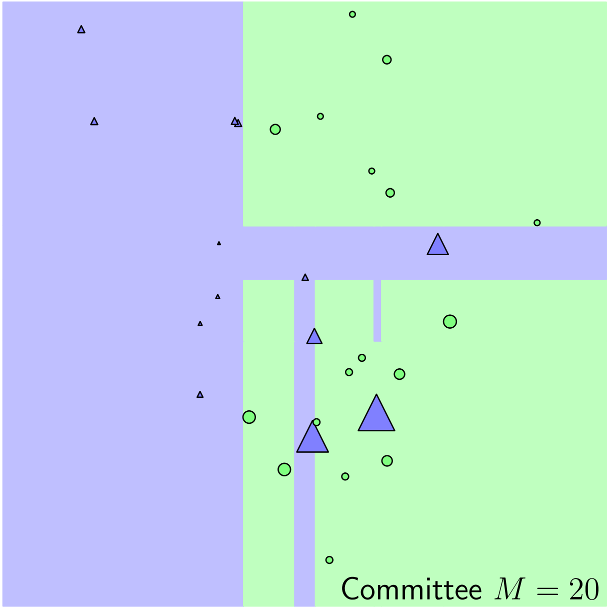

Classifier 6

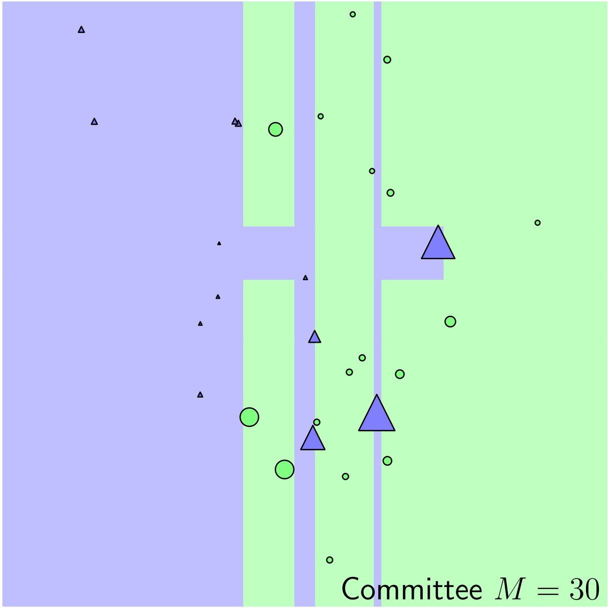

Committee $M = 6$


Classifier 7


Committee $M = 7$


Committee $M = 10$


Committee $M = 20$


Committee $M = 30$

About Adaboost

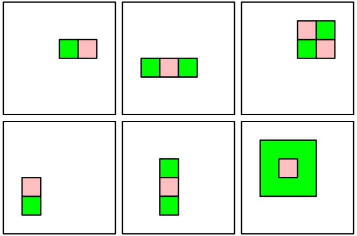
Adaboost can be interpreted as minimising

$$E = \sum_{n=1}^{N} \exp\left(-\frac{t_n}{2}\sum_{m=1}^{M}\alpha_m y_m(\mathbf{x}_n)\right)$$

As a consequence:

1. It strongly penalises misclassifications, not robust to outliers!
2. It does not generalise to more than 2 classes
3. Choosing a different error function
   - Allows multiclass classification and even regression (e.g. Gradient Boosting)
   - Makes robust classifiers possible

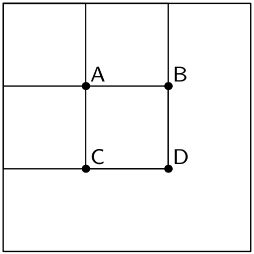
Viola-Jones face detector



A nice application of boosting:

- Very simple features (HAAR wavelets)
  - Use integral images to compute these very fast
- Use *cascading* for speedup

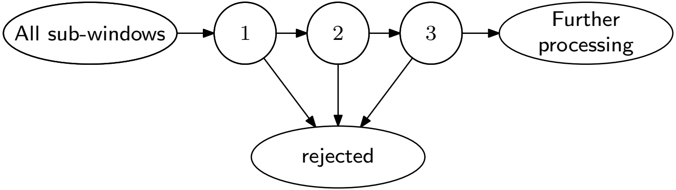
Viola-Jones face detector



A nice application of boosting:

- Very simple features (HAAR wavelets)
  - Use integral images to compute these very fast
- Use *cascading* for speedup

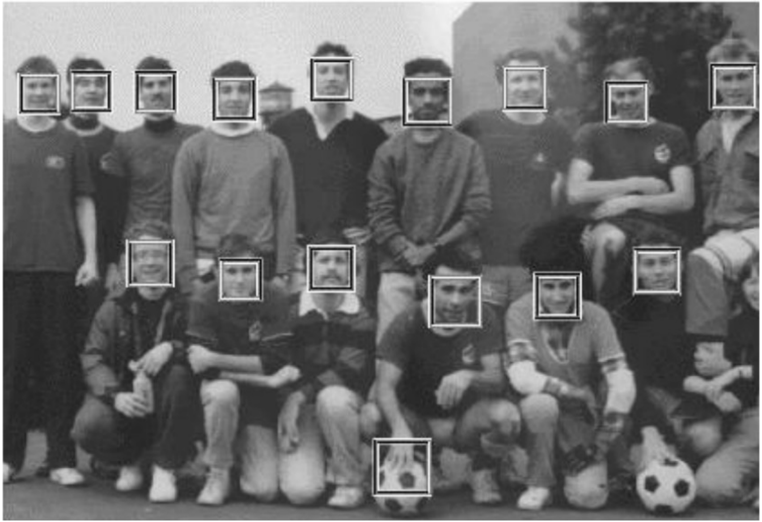
Viola-Jones face detector



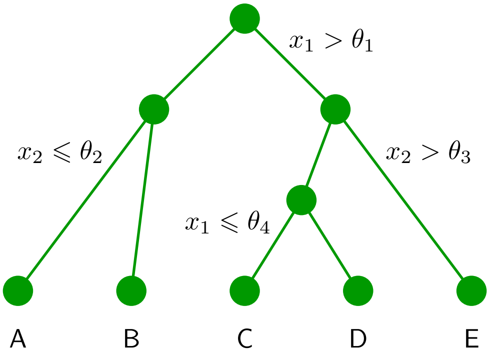
A nice application of boosting:

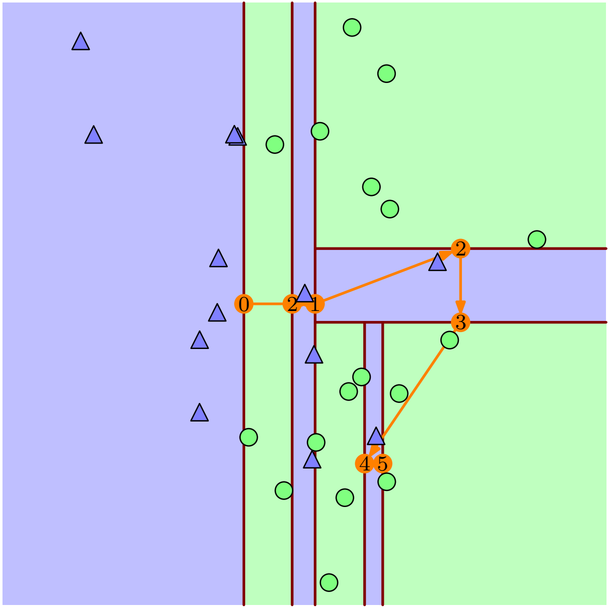
- Very simple features (HAAR wavelets)
  - Use integral images to compute these very fast
- Use *cascading* for speedup

Viola-Jones face detector

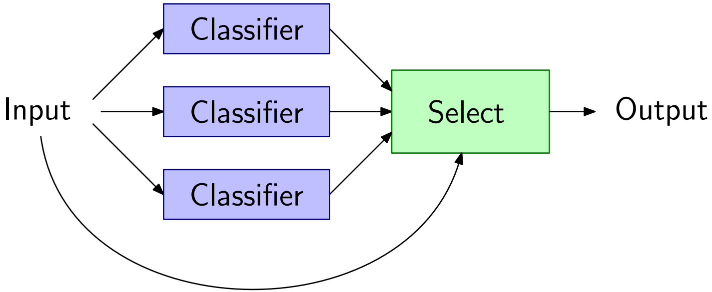


Binary Tree classifier




0
2
1
2
3
4
5

Tree-based models



Tree-based models split the input space in regions

- Each region gets its own classifier
- The classifiers can be extremely simple (typically: constant)

Tree-based models

Pros	Cons
- Interpretable! - Simple and fast - If let to grow, will learn perfect classification on the training data - Pruning (using validation set) allows proper generalisation	- Final tree depends strongly on particular data - Hard decisions, aligned with dimensions - Finding best tree is intractable

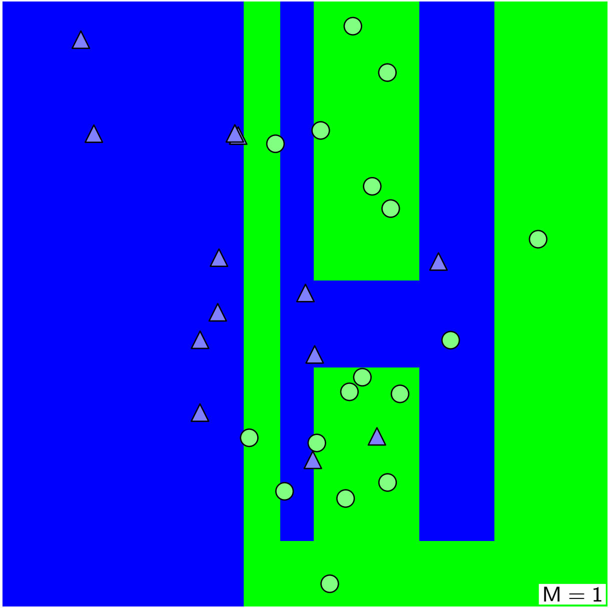
Random Forests

Combine trees with bagging and random feature selection
Procedure: for N datapoints and M features,
pre-specify $m \ll M$

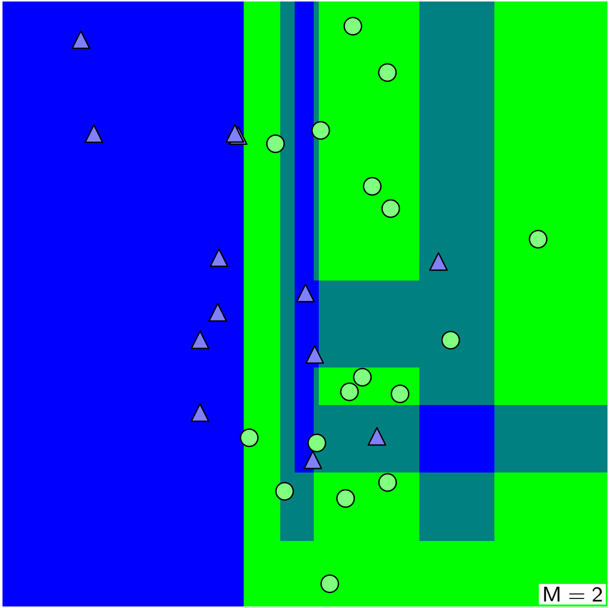
1. Repeat K times:
    1.1 Get a bootstrap sample
    1.2 At each node in the tree:
        1.2.1 select m features at random
        1.2.2 Find the optimal split based on these m features and the training set
    1.3 Fully grow the tree (no pruning)

This is often considered one of the most powerful committee methods

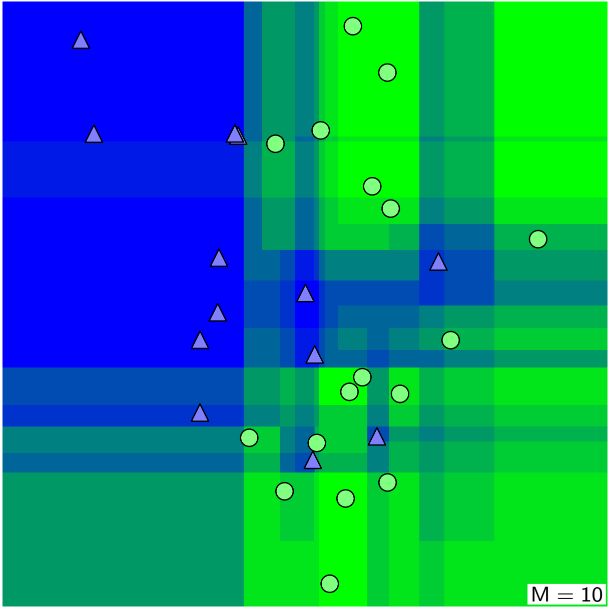
$M = 1$



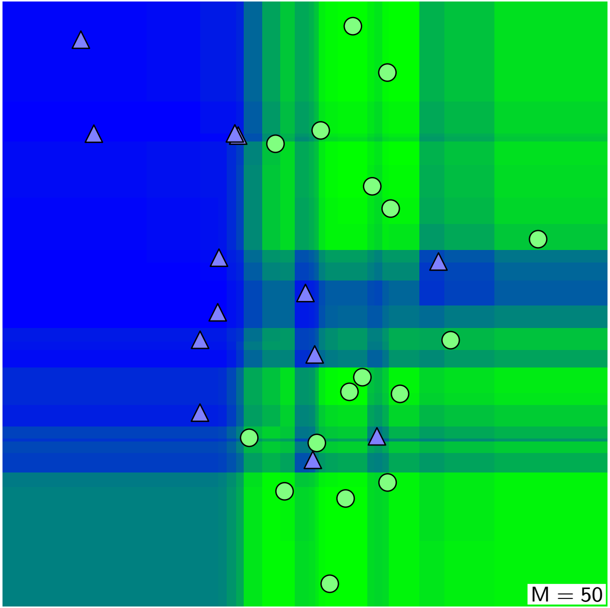
$M = 2$



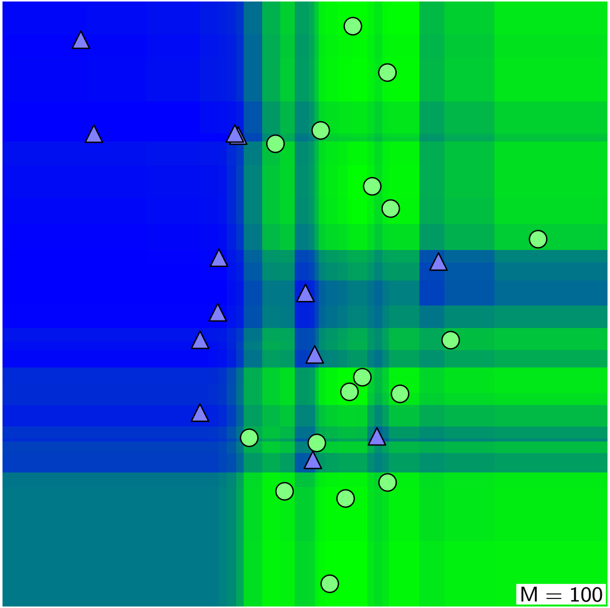
$M = 10$



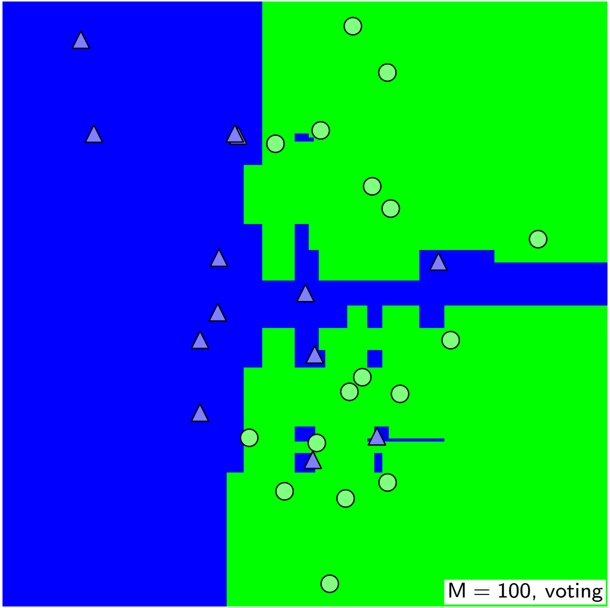
$M = 50$



$M = 100$



$M = 100$ Final decision



Wrap-up

To summarise:

- Combine models to improve their expressive power (cfr. Mixture of Gaussians)
- Combining independent models can dramatically improve performance
- Making different models responsible for different areas of the space combines simple models into very flexible models