Machine Learning
Boosting

Paul Swoboda

Lecture 15, 10.12.2018

Roadmap of today and next lecture

Classification methods:

- **Boosting**,
- Decision Trees,
- Neural Networks (aka Deep Learning),
- **Nearest Neighbor Methods, Parzen-Window**,

Roadmap of today

Boosting

- **Given:** a set of (bad) classifiers.
  Question: Is there a way to combine them so that they yield a reasonable classifier ?
- Boosting “boosts” existing bad/weak classifiers.
- Developed by Schapire and Freund (1996) - but concept is older.
- Final classification can often be implemented quite efficiently - interesting for real time application e.g. face detection on a video sequence.

Boosting belongs to the ensemble methods

History of Boosting

- Freund and Schapire propose Adaboost in 1996,
  Folklore result: Adaboost does not overfit !
- Friedman, Hastie and Tibshirani (2000) provide interpretation of Adaboost in terms of mimimization of empirical exponential loss
  $\Longrightarrow$ provides a general scheme for boosting which leads to emergence of new variants.
- Lugosi and Vayatis (2004) prove for a regularized boosting variant that it is Bayes consistent.
- Bartlett and Traskin (2007) prove that Adaboost is Bayes consistent.
- still ongoing discussion about interpretation after more than 10 years of boosting (recent controversial JMLR article).

Boosting II

General scheme for boosting methods

1. For each training point (X_i, Y_i) one has a weight γ_i.
2. A step of boosting method involves the following steps
    1. one trains a classifier f_k using a base method (weak learner) with the weighted training data (X_i, Y_i, γ_i),
    2. one re-computes the weights γ, where usually the weights of wrongly classified training points are increasing and the weights of correctly classified points are decreasing.
3. One aggregates the classifiers f_k to the final classifier $F(x) = \mathrm{sign}(\sum_{k=1} \alpha_k f_k)$, where the coefficients α_k are either one or depend on the error of the classifier f_k.

Variants of Boosting

Caution

- Several boosting methods have been proposed (huge literature),
- Differences are often very subtle,
- Different requirements on the properties of the weak learner.

Note: Boosting heavily depends on the weak learner
$\Longrightarrow$ two boosting methods with different weak learner cannot be compared !

Today: **Adaboost** and **GentleBoost**.

Properties

- Adaboost stands for "Adaptive Boosting"
- Many variants - we discuss Adaboost.M1 proposed by Freund and Schapire in 1996.
- Weak learner has to be **binary-valued** !
- Depending on the weak learner - final classifier allows interpretation (boosted decision stumps).

Adaboost

Input: Training data $(X_i, Y_i)_{i=1}^n$, binary-valued Weak Learner, Number of iterations T.
Initialize weights: $\gamma_i^1 = \frac{1}{n}$, $i = 1, \ldots, n$ For $t = 1, \ldots, T$, 1) Fit the weak learner, $f_t : \mathcal{X} \to \{-1, 1\}$, with weights γ_i. The weak learner uses the weighted zero-one loss $L(f) = \sum_{i=1}^n \gamma_i \mathbb{1}_{f(X_i) \neq Y_i}$ 2) compute the weighted error of f_t, $\mathrm{err}_t = \sum_{i=1}^n \gamma_i \mathbb{1}_{f(X_i) \neq Y_i}$, 3) define $c_t = \log\left(\frac{1-\mathrm{err}_t}{\mathrm{err_t}}\right)$, 4) update the weights γ_i^t, $\gamma_i^{t+1} = \gamma_i^t \exp(c_t \mathbb{1}_{Y_i \neq f_t(X_i)})$, 5) renormalize so that $\sum_{i=1}^n \gamma_i^{t+1} = 1$.
Output: final classifier $f(x) = \mathrm{sign}(\sum_{t=1}^T c_t f_t(x))$.

Adaboost II

Idea of Adaboost:

- in each step: fit weak learner to data,
- data which is misclassified gets higher weight
  $\Longrightarrow$ next weak learner will try harder to fit misclassified points,
- contribution c_t of each weak classifier f_t to the final hypothesis,

$$c_t = \log\left(\frac{1 - \mathrm{err}_t}{\mathrm{err}_t}\right)$$

  $\Longrightarrow$ strictly monotonically decreasing with err_t.
- final classifier: $f(x) = \mathrm{sign}\left(\sum_{t=1}^{T} c_t f_t(x)\right)$,
  $\Longrightarrow$ Adaboost learns a point in the vector space of functions generated by the weak learner.

Adaboost in action

Decision stump as weak learner:

$$f_t(x) = 2\mathbb{1}_{x_i+b>0} - 1 \quad \text{and} \quad f_t(x) = 2\mathbb{1}_{-x_i+b>0} - 1.$$

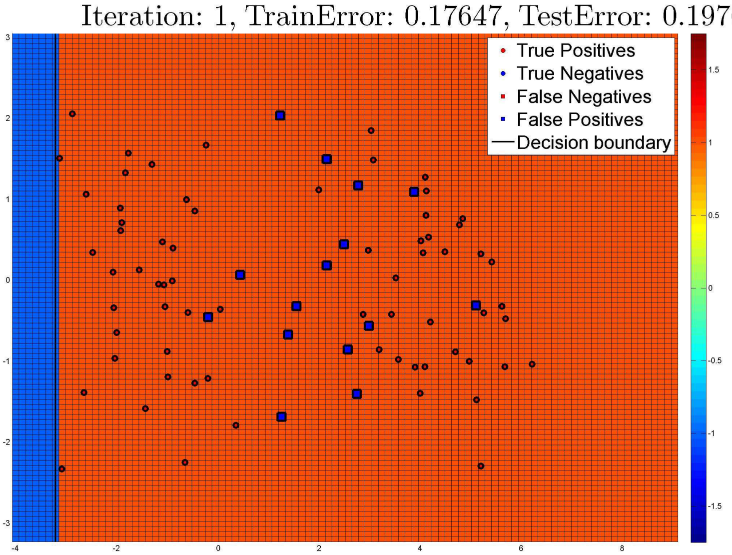
or more generally,

$$f_t(x) = 2\mathbb{1}_{\langle w,x\rangle+b>0} - 1.$$

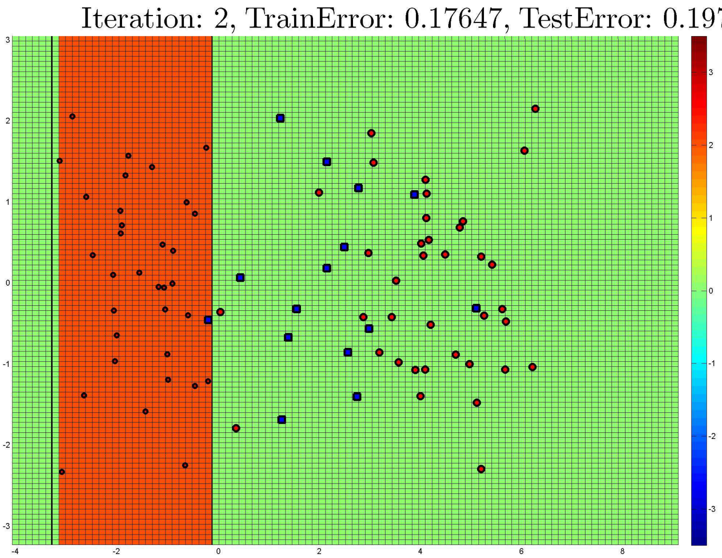
Selection of the weak learner:

- random (e.g. randomly select coordinate or direction w)
- compute for all choices weak learner (if possible) and take the one with the smallest weighted error.

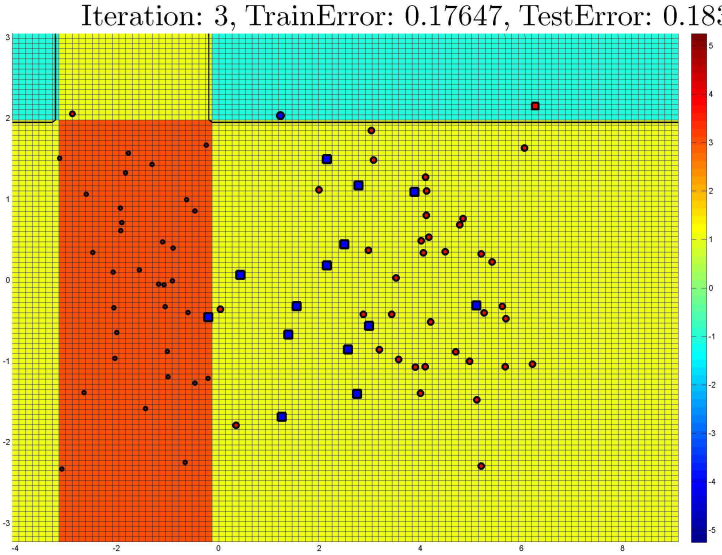
Adaboost in action



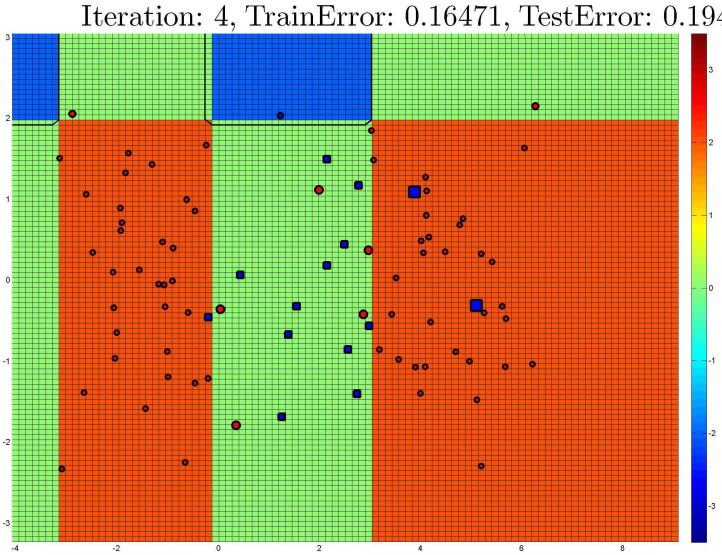
Adaboost in action



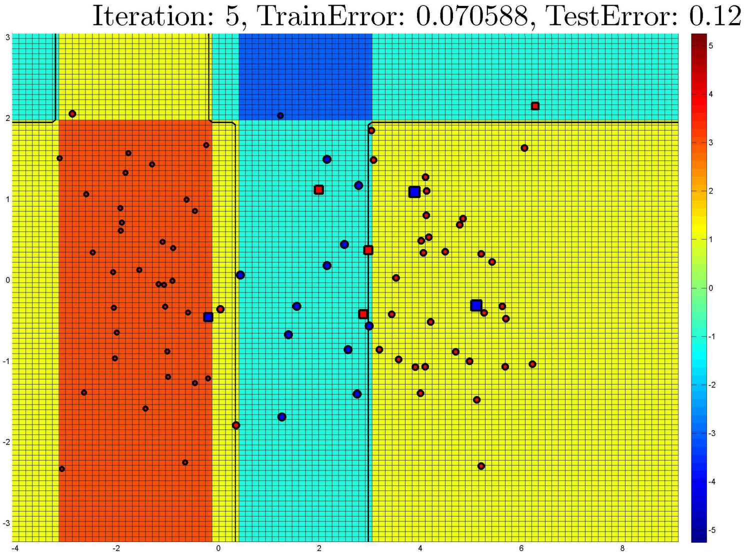
Adaboost in action



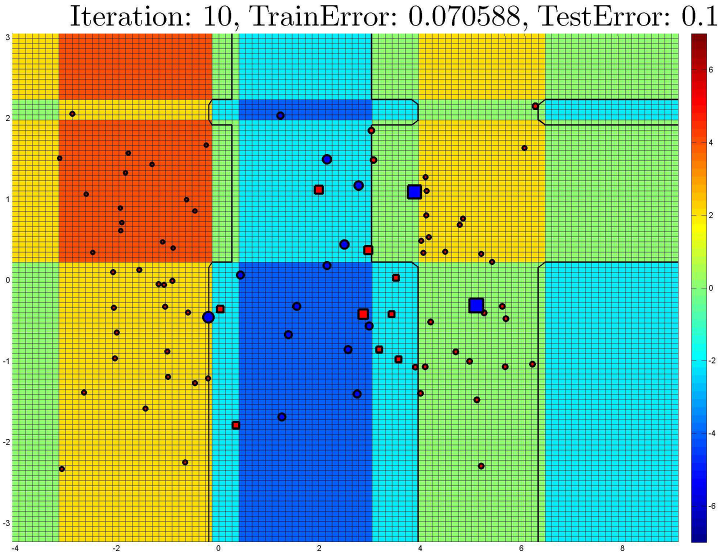
Adaboost in action



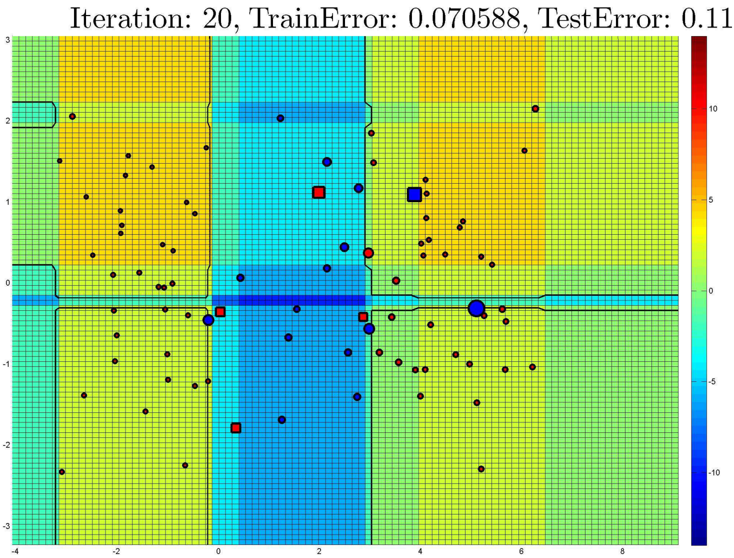
Adaboost in action



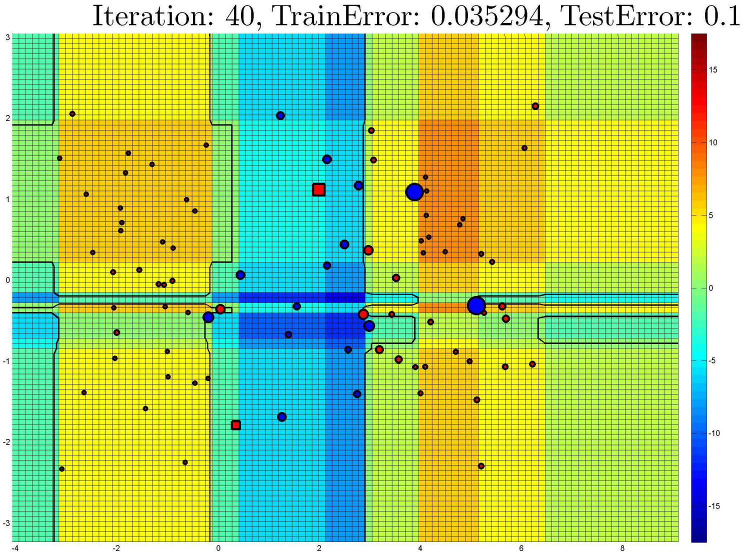
Adaboost in action



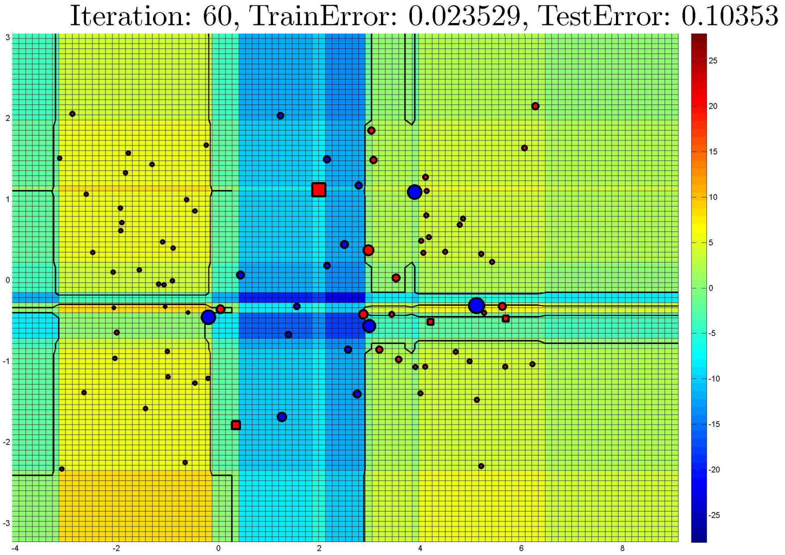
Adaboost in action



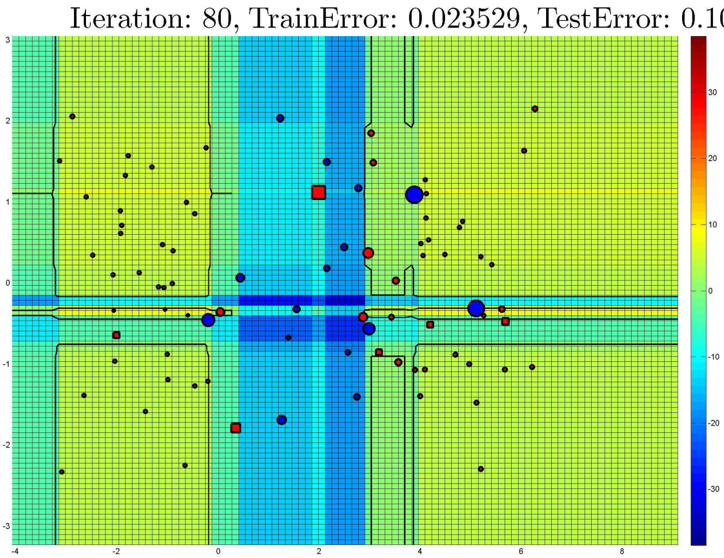
Adaboost in action



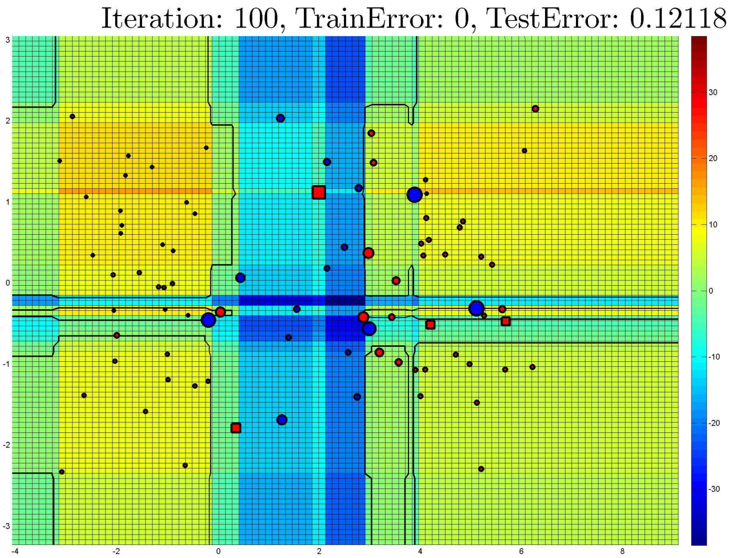
Adaboost in action



Adaboost in action



Adaboost in action



Adaboost in action

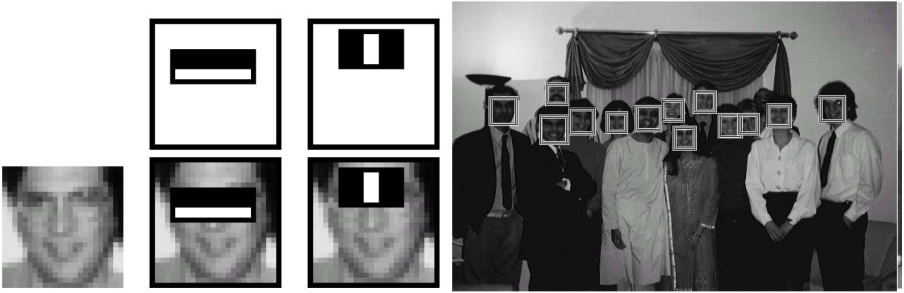
Adaboost in Computer Vision

- final classification using decision stump can be done extremely fast $\Longrightarrow$ very interesting for applications where real-time performance is required.
- Most successful application of Adaboost in face recognition by Viola and Jones (2001).
  They use more generally weak learner of the form,

  $$f_t(x) = 2\mathbb{1}_{\langle w,x\rangle + b > 0} - 1$$

  but coefficients w are restricted to be binary valued or zero (most coefficients of the weight vector w are zero).

Face Recognition



Interpretation of boosting

- Iterative updates,
$$F(x) \;\mapsto\; F(x) + c_t f_t(x),$$
can be understood as a descent in function space, where
  1. f_t is a descent direction based on the current F,
  2. c_t is the stepsize of the descent step.
- The objective which is minimized is the empirical exponential loss of the final classifier F,
$$L(F) = \frac{1}{n}\sum_{i=1}^{n} e^{-Y_i F(X_i)}.$$

Interpretation of boosting

- initiated simultaneously in the statistics and machine learning community,
- following Proposition is a variation of the result of Friedman, Hastie and Tibshirani (2000).

Proposition

Suppose the weak learner f_t is binary-valued, $f_t : \mathcal{X} \to \{-1, 1\}$. The update step $F_{t+1} = F_t + c_t f_t$ of the Adaboost algorithm , where $c_t = \log\left(\frac{1-\mathrm{err}_t}{\mathrm{err}_t}\right)$ and err_t is the weighted zero-one loss, is a descent step in order to minimize the empirical exponential loss.

$\Rightarrow$ proof on blackboard !

Result for Adaboost

Corollary

After each update of the weights, the weighted misclassification error of the most recent weak learner is 50%.

Proof: This follows directly from the optimality condition for the parameter c derived in the last proof,

$$\mathbb{E}[Y\, f(X)\, e^{-Y\, F(X)}\, e^{-Y\, f(X)\alpha}] = 0.$$

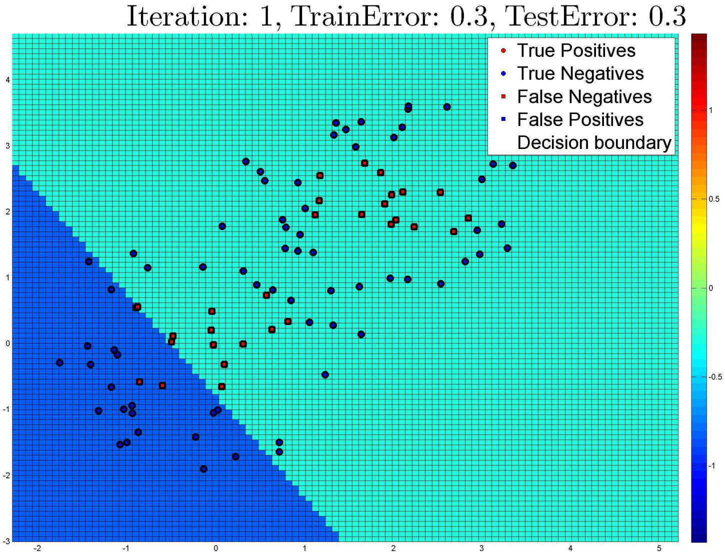
Note, that the second factor corresponds to the new weights (up to normalization) and the weighted zero one loss of f is given as

$$\frac{\frac{1}{2}\mathbb{E}[(1 - Y\, f(X))e^{-Y\, F(X)}\, e^{-Y\, f(X)\alpha}]}{\mathbb{E}[e^{-Y\, F(X)}\, e^{-Y\, f(X)\alpha}]} = \frac{1}{2}.$$

Theory has motivated many variants of boosting $\Longrightarrow$ **GentleBoost**

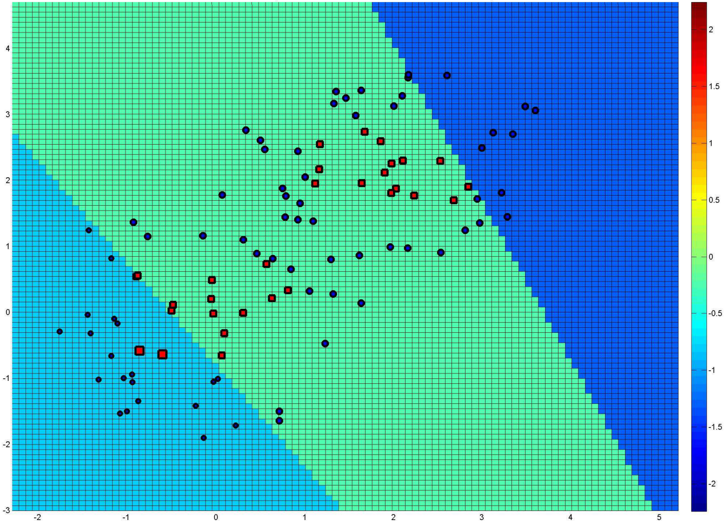
Input: Training data $(X_i, Y_i)_{i=1}^n$, real-valued Weak Learner, Number of iterations T.
Initialize weights: $\gamma_i^1 = \frac{1}{n}$, $i = 1, \dots, n$. For $t = 1, \dots, T$, 1) Fit the weak learner, $f_t : \mathcal{X} \to \mathbb{R}$, with weights γ_i. The weak learner uses weighted squared loss $L(f) = \sum_{i=1}^n \gamma_i (Y_i - f_t(X_i))^2$, 2) update the weights γ_i^t, $\gamma_i^{t+1} = \gamma_i^t \exp(-Y_i f_t(X_i))$, 5) renormalize so that $\sum_{i=1}^n \gamma_i^{t+1} = 1$.
Output: final classifier $f(x) = \mathrm{sign}\left(\sum_{t=1}^T f_t(x)\right)$.

GentleBoost in action

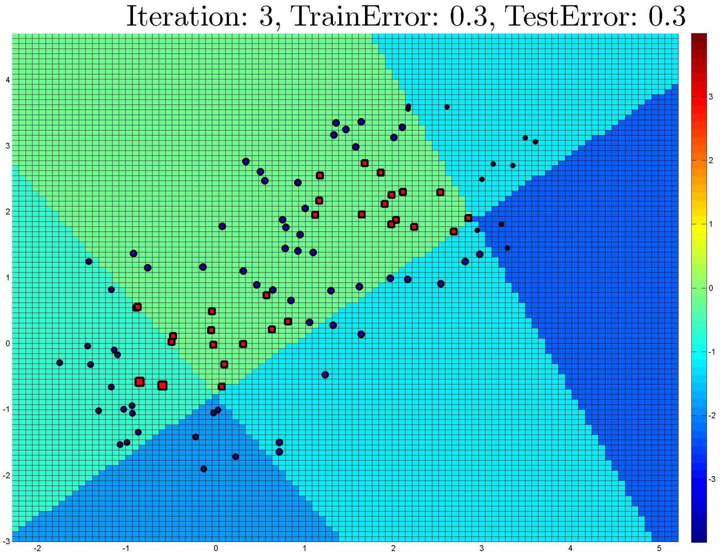


GentleBoost in action

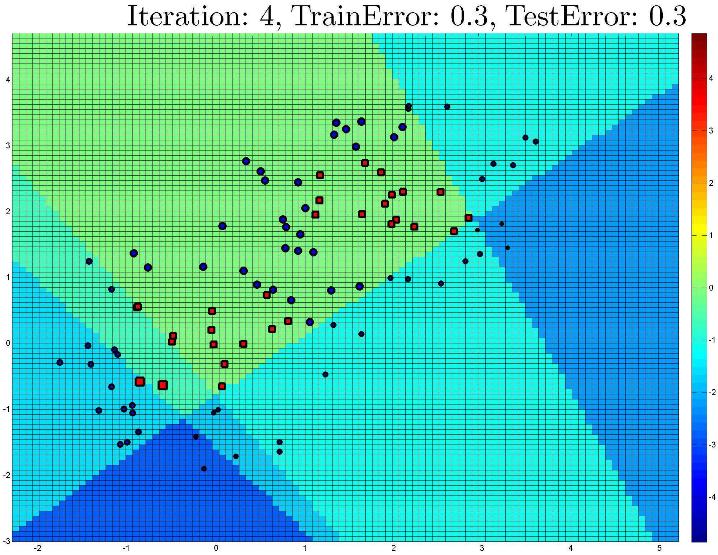
Iteration: 2, TrainError: 0.3, TestError: 0.3



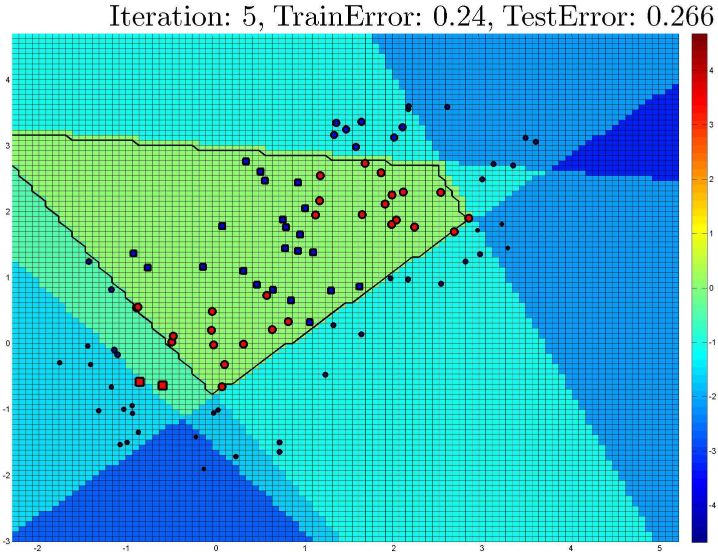
GentleBoost in action



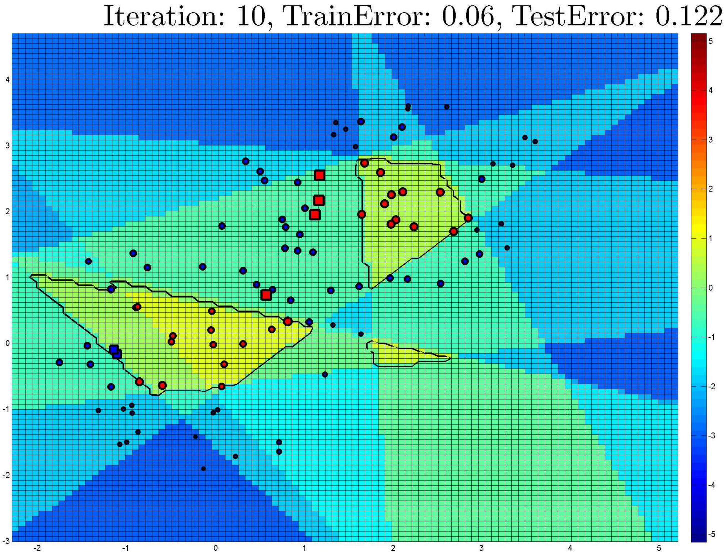
GentleBoost in action



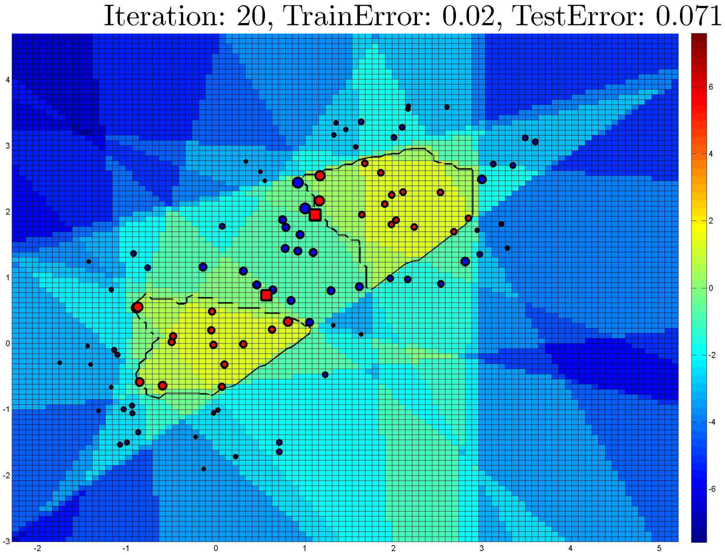
GentleBoost in action



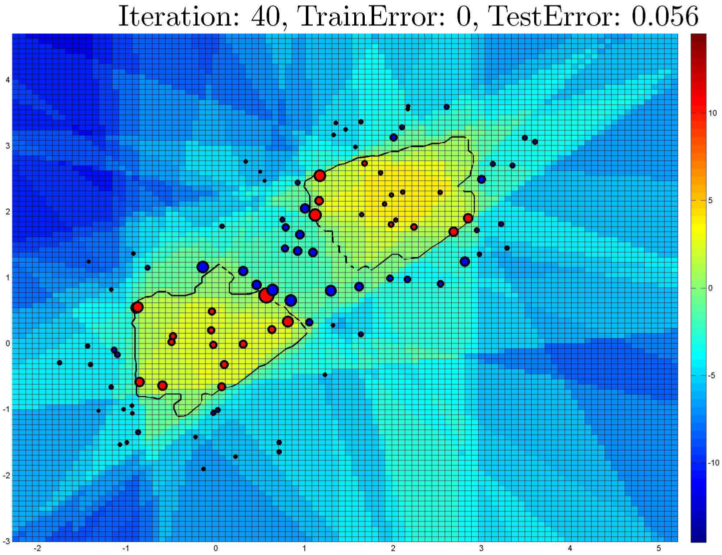
GentleBoost in action



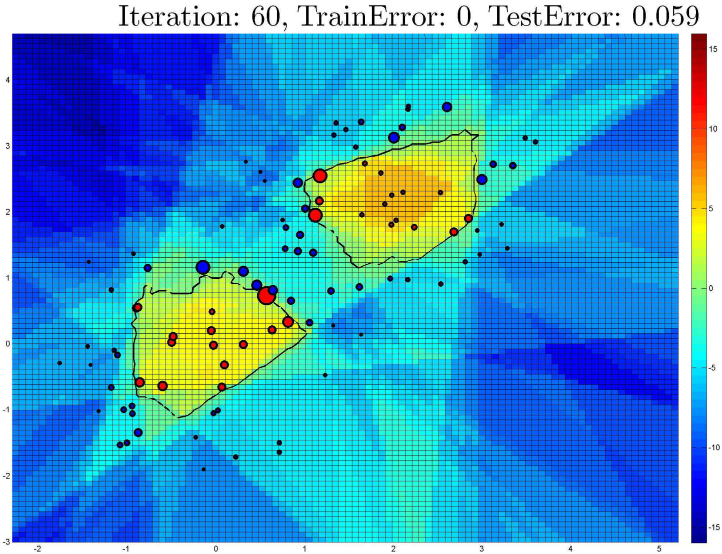
GentleBoost in action



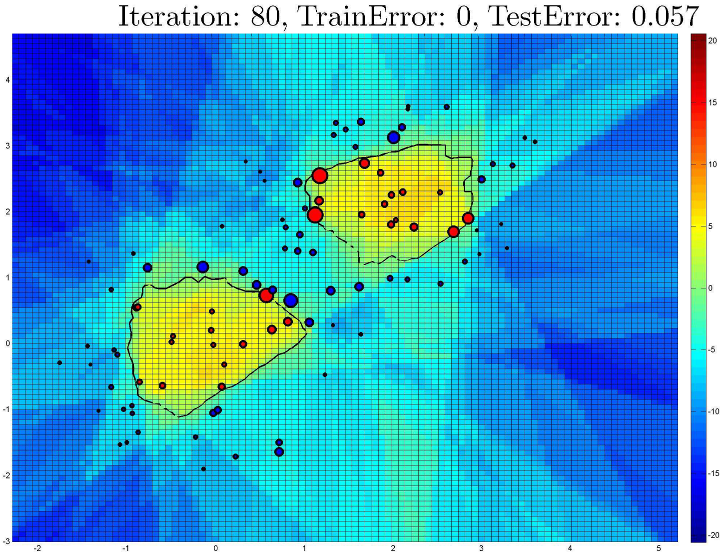
GentleBoost in action



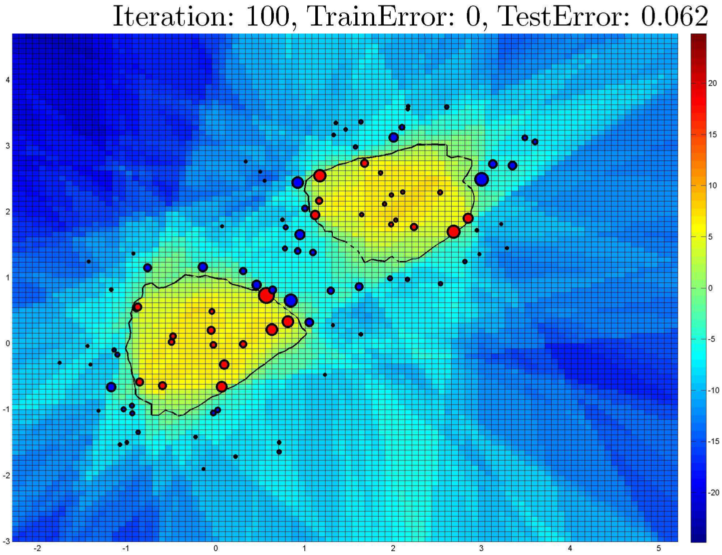
GentleBoost in action



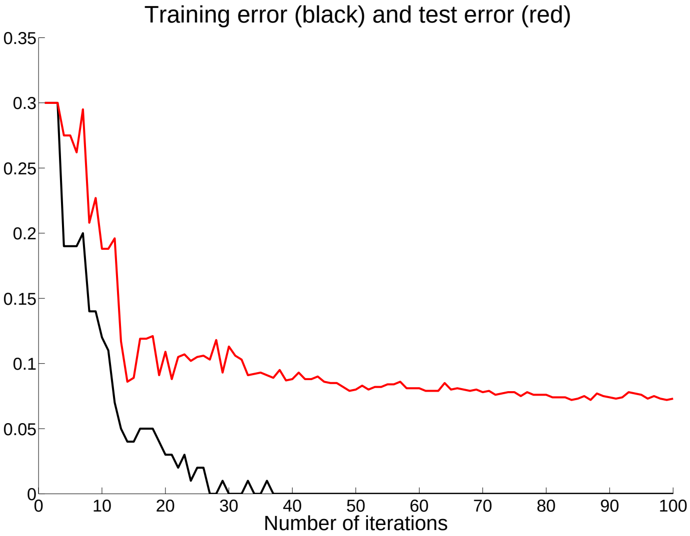
GentleBoost in action



GentleBoost in action



GentleBoost in action



Proposition

Suppose the weak learner f_t is real-valued, $f_t : \mathcal{X} \to \mathbb{R}$. The update step $F_{t+1} = F_t + f_t$ of the GentleBoost algorithm is an approximate Newton step in order to minimize the empirical exponential loss.

Proof: We can expand the risk of $F + f$ up to second order,

$$R(F+f) = \mathbb{E}[e^{-Y(F(X)+f(X))}] \approx \mathbb{E}[e^{-Y F(X)}(1 - Y f(X) + \frac{1}{2} f(X)^2)].$$

The first term does not depend on f so we can modify it without changing the minimizer,

$$\mathbb{E}[e^{-Y F(X)}(\frac{1}{2} - Y f(X) + \frac{1}{2} f(X)^2)] = \frac{1}{2}\mathbb{E}[e^{-Y F(X)}(Y - f(X))^2],$$

which is up to the normalization of the weights $\gamma_i = e^{-Y_i F(X_i)}$ equal to the weighted squared loss minimized by the weak learner $\Rightarrow$ weak learner minimizes in each step a second order approximation of the exponential loss.