Learning:
Boosting and Adaboost

http://www.cs.ucsd.edu/~yfreund/adaboost/index.html
Combining Weak learners
- Empirical risk and the overfitting dilemma
- Weak learners
- The need of combining learners
- Bagging

The Adaboost Algorithm
- Boosting vs Bagging
- Adaboost
- Example
- Additional remarks
- Evaluation: Cross-validation
Combining Weak learners
- Empirical risk and the overfitting dilemma
- Weak learners
- The need of combining learners
- Bagging

The Adaboost Algorithm
- Boosting vs Bagging
- Adaboost
- Example
- Additional remarks
- Evaluation: Cross-validation
References


- www.boosting.org
Empirical risk

- **Reviewing classifiers:**
  1. **Patterns:** \( x_i \in X = \mathbb{R}^d \)
  2. **Classes:** \( y_i \in Y = \{-1, 1\} \)
  3. **Training set:**
     \[(x_1, y_1), \ldots, (x_N, y_N)\]
  4. **Learned function:**
     \[f : \mathbb{R}^d \rightarrow Y\]
  5. **Classifier:** \( h_f(x) = \text{sign}(f(x)) \)

- **Empirical risk:**
  - A good classifier should minimize the average disparity wrt supervision.
  - Considers only the training set.

\[
L(f) = \frac{1}{N} \sum_{i=1}^{N} I(h_f(x_i) \neq y_i)
\]
The overfitting dilemma

- **Complexity of “f”?**
- **Overfitting dilemma:** Increasing the complexity of “f” (for instance to overcome non-linear separability) reduces the empirical risk but leads to very poor generalization (overfitting)
- **Occam’s Razor:** The simple function explaining most of the data is preferable to a complex one fitting the data very well.
- **Example:** If you have a Multi-layer perceptron with many hidden units it will overfit, that is, memorize the training data.
Combine weak learners?

- **Weak learners:** Moderately accurate (simple) classifiers (working better than by chance). The result is an ensemble hypothesis.

- **Combination:**
  - It has been proved that it is possible to find a more accurate classifier by combining many weak classifiers.
  - How to combine the weak classifiers?
    - Bagging
    - Boosting
Bagging. [Breiman, 94] Repeat for $t = 1, \ldots, T$:

- Select, at random with replacement, $N$ training examples.
- Train learner on selected samples to generate $h_t$

Final hypothesis is simple vote:

$$H(x) = \text{MAJ}(h_1(x), \ldots, h_T(x))$$

- Features:
  - Helps improving unstable classifiers, like NNs or trees (small changes in training data lead to different classifiers and large changes in accuracy).
Combining Weak learners
- Empirical risk and the overfitting dilemma
- Weak learners
- The need of combining learners
- Bagging

The Adaboost Algorithm
- Boosting vs Bagging
- Adaboost
- Example
- Additional remarks
- Evaluation: Cross-validation
Weighthed sampling (examples):
- Instead of random sampling of the training data, samples are weighted in order to focus learning on most difficult examples.
- Intuitively, examples near the decision boundary are harder to classify and will receive higher weights.

Weighthed vote (classifiers):
- Instead of combining classifiers with equal vote, a weighted vote is used.
- This is combination rule for the ensemble of weak learners.
- In conjunction with the latter sampling strategy this produces a stronger classifier.
Boosting intuition

1st Iteration

2nd Iteration

3rd Iteration

5th Iteration

10th Iteration

100th Iteration

Combined Sigle

Bagging
**AdaBoost. Adaptive Boosting** [Freund, Schapire, 96]

- Initialize distribution over training set $D_1(i) = 1/N$.
- For $t = 1, \ldots, T$
  1. Train weak learner using distribution $D_t$ and obtain $h_t$.
  2. Choose a weight (confidence value) $\alpha_t \in R$.
  3. Update distribution over training set:
     \[
     D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_t}
     \]

Set $H(x) = sign(f(x)) = sign \left( \sum_{t=1}^{T} \alpha_t h_t(x) \right)$

**Remarks:**

- $i$ indexes examples, whereas $t$ indexes classifiers (weak learners)
- $D_t$ depends on the complexity of the examples. How to use it?
- $\alpha_t$ depends on the error $\varepsilon_t$ associated to the $h_t$
- $Z_t$ is a normalization constant.
Using and building Dt

1. Train a weak learner using $D_t$ and obtain $h_t$
   - Usually sample the training examples using $D_t$ (importance sampling)
   - When $T=1$ all examples are equally probable.
   - Later, more difficult examples (those making the classifier fail) are more probable to be chosen.

2. Choose a confidence value $\alpha_t$
   - Let $\varepsilon_t$ the error associated to $h_t$
     \[
     \varepsilon_t = Pr_{D_t}[h_t(x_i) \neq y_i]
     \]
   - Then $\alpha_t$ comes after optimization and it is:
     \[
     \alpha_t = \frac{1}{2} \left( \frac{1 - \varepsilon_t}{\varepsilon_t} \right) > 0
     \]
3. Update distribution D:
   - When T=1 all examples are equally probable.
   - Later, more difficult examples (those making the classifier fail) are more probable to be chosen.

\[ D_{t+1}(i) = \frac{D_t(i)}{Z_t} \cdot A \]

\[ i f \quad h_t(x_i) = y_i \implies A = e^{-\alpha t} \]
\[ i f \quad h_t(x_i) \neq y_i \implies A = e^{\alpha t} \]
Example

Round 1

\[ D_1 \]

\[ D_2 \]

\[ \varepsilon_1 = 0.30 \]

\[ \alpha_1 = 0.42 \]
Example

Round 2

$\varepsilon_2 = 0.21$

$\alpha_2 = 0.65$

$D_3$
Example

Round 3

$h_3$

$\varepsilon_3 = 0.14$
$\alpha_3 = 0.92$
Example

**Final Hypothesis**

\[
H_{\text{final}} = \text{sign} \left( 0.42 \right) + 0.65 + 0.92
\]

= 

\[
\begin{array}{ccc}
  + & - & - \\
  + & - & - \\
  + & - & - \\
\end{array}
\]
Additional remarks

- **Probabilistic interpretation:** Posterior given the data (Yuille):

  \[
  P(y = 1|x) = \frac{e^{\sum_t \alpha_t h_t(x)}}{e^{\sum_i \alpha_t h_t(x)} + e^{-\sum_i \alpha_t h_t(x)}}
  \]

  \[
  P(y = -1|x) = \frac{e^{-\sum_t \alpha_t h_t(x)}}{e^{\sum_i \alpha_t h_t(x)} + e^{-\sum_i \alpha_t h_t(x)}}
  \]

- **Warnings:**
  - As we add weak classifiers we may MEMORIZE (not learn)
  - It is important to perform Cross Validation

  **K-Fold Cross Validation**
Evaluation: Cross Validation

- **Validation set:**
  - Randomly split the set of labeled training examples into two parts.
  - One part is used as a traditional training set for adjusting model parameters in the classifier.
  - The other part is the validation set and it is used to estimate the generalization error.
  - Train until reaching a minimum of the validation error.

- **K-fold Cross Validation:**
  - Randomly divide the training set into $K$ disjoint sets of equal size $N/K$ being $N$ the total number of training examples.
  - Train the classifier $K$ times, each time with a different set as a validation set.
  - The estimated performance is the mean of the $\eta$ errors.
  - When $N \approx K$ we have the Leaving-one-out