

RWTH AACHEN
UNIVERSITY

Machine Learning – Lecture 10

AdaBoost

20.11.2017

Bastian Leibe
RWTH Aachen
<http://www.vision.rwth-aachen.de>
leibe@vision.rwth-aachen.de

Machine Learning Winter '17

RWTH AACHEN
UNIVERSITY

Course Outline

- Fundamentals
 - Bayes Decision Theory
 - Probability Density Estimation
- Classification Approaches
 - Linear Discriminants
 - Support Vector Machines
 - Ensemble Methods & Boosting
 - Random Forests
- Deep Learning
 - Foundations
 - Convolutional Neural Networks
 - Recurrent Neural Networks

B. Leibe

RWTH AACHEN
UNIVERSITY

Recap: SVM – Analysis

- Traditional soft-margin formulation

$$\min_{\mathbf{w} \in \mathbb{R}^D, \xi_n \in \mathbb{R}^+} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{n=1}^N \xi_n$$

"Maximize the margin"

subject to the constraints

$$t_n y(\mathbf{x}_n) \geq 1 - \xi_n$$

"Most points should be on the correct side of the margin"
- Different way of looking at it
 - We can reformulate the constraints into the objective function.
$$\min_{\mathbf{w} \in \mathbb{R}^D} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{n=1}^N [1 - t_n y(\mathbf{x}_n)]_+$$

$\frac{1}{2} \|\mathbf{w}\|^2$ $C \sum_{n=1}^N [1 - t_n y(\mathbf{x}_n)]_+$
 L_2 regularizer "Hinge loss"

where $[x]_+ := \max\{0, x\}$.

B. Leibe

RWTH AACHEN
UNIVERSITY

Recap: Bayesian Model Averaging

- Model Averaging
 - Suppose we have H different models $h = 1, \dots, H$ with prior probabilities $p(h)$.
 - Construct the marginal distribution over the data set
$$p(\mathbf{X}) = \sum_{h=1}^H p(\mathbf{X}|h)p(h)$$
- Average error of committee

$$\mathbb{E}_{COM} = \frac{1}{M} \mathbb{E}_{AV}$$
 - This suggests that the average error of a model can be reduced by a factor of M simply by averaging M versions of the model!
 - Unfortunately, this assumes that the errors are all uncorrelated. In practice, they will typically be highly correlated.

B. Leibe

RWTH AACHEN
UNIVERSITY

Topics of This Lecture

- AdaBoost
 - Algorithm
 - Analysis
 - Extensions
- Analysis
 - Comparing Error Functions
- Applications
 - AdaBoost for face detection
- Decision Trees
 - CART
 - Impurity measures, Stopping criterion, Pruning
 - Extensions, Issues
 - Historical development: ID3, C4.5

B. Leibe

RWTH AACHEN
UNIVERSITY

AdaBoost – "Adaptive Boosting"

- Main idea [Freund & Schapire, 1996]
 - Iteratively select an ensemble of component classifiers
 - After each iteration, reweight misclassified training examples.
 - Increase the chance of being selected in a sampled training set.
 - Or increase the misclassification cost when training on the full set.
- Components
 - $h_m(\mathbf{x})$: "weak" or base classifier
 - Condition: <50% training error over any distribution
 - $H(\mathbf{x})$: "strong" or final classifier
- AdaBoost:
 - Construct a strong classifier as a thresholded linear combination of the weighted weak classifiers:
$$H(\mathbf{x}) = \text{sign} \left(\sum_{m=1}^M \alpha_m h_m(\mathbf{x}) \right)$$

B. Leibe

Machine Learning Winter '17

AdaBoost: Intuition

Consider a 2D feature space with **positive** and **negative** examples.

Each weak classifier splits the training examples with at least 50% accuracy.

Examples misclassified by a previous weak learner are given more emphasis at future rounds.

Slide credit: Kristen Grauman B. Leibe Figure adapted from Freund & Schapire 8

Machine Learning Winter '17

AdaBoost: Intuition

Weights Increased

Weak Classifier 2

Slide credit: Kristen Grauman B. Leibe Figure adapted from Freund & Schapire 9

Machine Learning Winter '17

AdaBoost: Intuition

The final classifier is a linear combination of the weak classifiers

Slide credit: Kristen Grauman B. Leibe Figure adapted from Freund & Schapire 10

Machine Learning Winter '17

AdaBoost – Formalization

- 2-class classification problem
 - Given: training set $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ with target values $\mathbf{T} = \{t_1, \dots, t_N\}$, $t_n \in \{-1, 1\}$.
 - Associated weights $\mathbf{W} = \{w_1, \dots, w_N\}$ for each training point.
- Basic steps
 - In each iteration, AdaBoost trains a new weak classifier $h_m(\mathbf{x})$ based on the current weighting coefficients $\mathbf{W}^{(m)}$.
 - We then adapt the weighting coefficients for each point
 - Increase w_n if \mathbf{x}_n was misclassified by $h_m(\mathbf{x})$.
 - Decrease w_n if \mathbf{x}_n was classified correctly by $h_m(\mathbf{x})$.
 - Make predictions using the final combined model

$$H(\mathbf{x}) = \text{sign} \left(\sum_{m=1}^M \alpha_m h_m(\mathbf{x}) \right)$$

Slide credit: Kristen Grauman B. Leibe 11

Machine Learning Winter '17

AdaBoost – Algorithm

- Initialization: Set $w_n^{(1)} = \frac{1}{N}$ for $n = 1, \dots, N$.
- For $m = 1, \dots, M$ iterations
 - Train a new weak classifier $h_m(\mathbf{x})$ using the current weighting coefficients $\mathbf{W}^{(m)}$ by minimizing the weighted error function

$$J_m = \sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}) \neq t_n) \quad I(A) = \begin{cases} 1, & \text{if } A \text{ is true} \\ 0, & \text{else} \end{cases}$$
 - Estimate the weighted error of this classifier on \mathbf{X} :

$$\epsilon_m = \frac{\sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}) \neq t_n)}{\sum_{n=1}^N w_n^{(m)}}$$
 - Calculate a weighting coefficient for $h_m(\mathbf{x})$:

$$\alpha_m = ?$$
 - Update the weighting coefficients:

$$w_n^{(m+1)} = ?$$

How should we do this exactly?

Slide credit: Kristen Grauman B. Leibe 12

Machine Learning Winter '17

AdaBoost – Historical Development

- Originally motivated by Statistical Learning Theory
 - AdaBoost was introduced in 1996 by Freund & Schapire.
 - It was empirically observed that AdaBoost often tends not to overfit. (Breiman 96, Cortes & Drucker 97, etc.)
 - As a result, the margin theory (Schapire et al. 98) developed, which is based on loose generalization bounds.
 - Note: margin for boosting is *not* the same as margin for SVM.
 - A bit like retrofitting the theory...
 - However, those bounds are too loose to be of practical value.
- Different explanation [Friedman, Hastie, Tibshirani, 2000]
 - Interpretation as sequential minimization of an exponential error function ("Forward Stagewise Additive Modeling").
 - Explains why boosting works well.
 - Improvements possible by altering the error function.

Slide credit: Kristen Grauman B. Leibe 13

RWTH AACHEN UNIVERSITY

AdaBoost – Minimizing Exponential Error

- Exponential error function

$$E = \sum_{n=1}^N \exp \{-t_n f_m(\mathbf{x}_n)\}$$
 - where $f_m(\mathbf{x})$ is a classifier defined as a linear combination of base classifiers $h_l(\mathbf{x})$:

$$f_m(\mathbf{x}) = \frac{1}{2} \sum_{l=1}^m \alpha_l h_l(\mathbf{x})$$
- Goal
 - Minimize E with respect to both the weighting coefficients α_l and the parameters of the base classifiers $h_l(\mathbf{x})$.

14

RWTH AACHEN UNIVERSITY

AdaBoost – Minimizing Exponential Error

- Sequential Minimization
 - Suppose that the base classifiers $h_1(\mathbf{x}), \dots, h_{m-1}(\mathbf{x})$ and their coefficients $\alpha_1, \dots, \alpha_{m-1}$ are fixed.
 - Only minimize with respect to α_m and $h_m(\mathbf{x})$.

$$E = \sum_{n=1}^N \exp \{-t_n f_m(\mathbf{x}_n)\} \quad \text{with} \quad f_m(\mathbf{x}) = \frac{1}{2} \sum_{l=1}^m \alpha_l h_l(\mathbf{x})$$

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

$= \text{const.}$

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

15

RWTH AACHEN UNIVERSITY

AdaBoost – Minimizing Exponential Error

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

- Observation:
 - Correctly classified points: $t_n h_m(\mathbf{x}_n) = +1 \Rightarrow$ collect in \mathcal{T}_m
 - Misclassified points: $t_n h_m(\mathbf{x}_n) = -1 \Rightarrow$ collect in \mathcal{F}_m
- Rewrite the error function as

$$E = e^{-\alpha_m/2} \sum_{n \in \mathcal{T}_m} w_n^{(m)} + e^{\alpha_m/2} \sum_{n \in \mathcal{F}_m} w_n^{(m)}$$

$$= \left(e^{\alpha_m/2} \right) \sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}_n) \neq t_n)$$

16

RWTH AACHEN UNIVERSITY

AdaBoost – Minimizing Exponential Error

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

- Observation:
 - Correctly classified points: $t_n h_m(\mathbf{x}_n) = +1 \Rightarrow$ collect in \mathcal{T}_m
 - Misclassified points: $t_n h_m(\mathbf{x}_n) = -1 \Rightarrow$ collect in \mathcal{F}_m
- Rewrite the error function as

$$E = \left(e^{-\alpha_m/2} \sum_{n \in \mathcal{T}_m} w_n^{(m)} + e^{\alpha_m/2} \sum_{n \in \mathcal{F}_m} w_n^{(m)} \right)$$

$$= \left(e^{\alpha_m/2} - e^{-\alpha_m/2} \right) \sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}_n) \neq t_n) + e^{-\alpha_m/2} \sum_{n=1}^N w_n^{(m)}$$

17

RWTH AACHEN UNIVERSITY

AdaBoost – Minimizing Exponential Error

- Minimize with respect to $h_m(\mathbf{x})$: $\frac{\partial E}{\partial h_m(\mathbf{x}_n)} \stackrel{!}{=} 0$

$$E = \underbrace{\left(e^{\alpha_m/2} - e^{-\alpha_m/2} \right)}_{= \text{const.}} \sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}_n) \neq t_n) + \underbrace{e^{-\alpha_m/2} \sum_{n=1}^N w_n^{(m)}}_{= \text{const.}}$$
 - \Rightarrow This is equivalent to minimizing

$$J_m = \sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}) \neq t_n)$$
 (our weighted error function from step 2a) of the algorithm)
 - \Rightarrow We're on the right track. Let's continue...

18

RWTH AACHEN UNIVERSITY

AdaBoost – Minimizing Exponential Error

- Minimize with respect to α_m : $\frac{\partial E}{\partial \alpha_m} \stackrel{!}{=} 0$

$$E = \left(e^{\alpha_m/2} - e^{-\alpha_m/2} \right) \sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}_n) \neq t_n) + e^{-\alpha_m/2} \sum_{n=1}^N w_n^{(m)}$$

$$\left(\frac{1}{2} e^{\alpha_m/2} + \frac{1}{2} e^{-\alpha_m/2} \right) \sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}_n) \neq t_n) \stackrel{!}{=} \frac{1}{2} e^{-\alpha_m/2} \sum_{n=1}^N w_n^{(m)}$$

$$\text{weighted error } \epsilon_m := \frac{\sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}_n) \neq t_n)}{\sum_{n=1}^N w_n^{(m)}} = \frac{e^{-\alpha_m/2}}{e^{\alpha_m/2} + e^{-\alpha_m/2}}$$

$$\epsilon_m = \frac{1}{e^{\alpha_m} + 1}$$
 - \Rightarrow Update for the α coefficients:

$$\alpha_m = \ln \left\{ \frac{1 - \epsilon_m}{\epsilon_m} \right\}$$

19

RWTH AACHEN UNIVERSITY

AdaBoost – Minimizing Exponential Error

- Remaining step: update the weights
 - Recall that

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

This becomes $w_n^{(m+1)}$ in the next iteration.
 - Therefore

$$\begin{aligned} w_n^{(m+1)} &= w_n^{(m)} \exp \left\{ -\frac{1}{2} t_n \alpha_m h_m(\mathbf{x}_n) \right\} \\ &= \dots \\ &= w_n^{(m)} \exp \{ \alpha_m I(h_m(\mathbf{x}_n) \neq t_n) \} \end{aligned}$$

⇒ Update for the weight coefficients.

20

RWTH AACHEN UNIVERSITY

AdaBoost – Final Algorithm

- Initialization: Set $w_n^{(1)} = \frac{1}{N}$ for $n = 1, \dots, N$.
- For $m = 1, \dots, M$ iterations
 - Train a new weak classifier $h_m(\mathbf{x})$ using the current weighting coefficients $\mathbf{W}^{(m)}$ by minimizing the weighted error function

$$J_m = \sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}) \neq t_n)$$
 - Estimate the weighted error of this classifier on \mathbf{X} :

$$\epsilon_m = \frac{\sum_{n=1}^N w_n^{(m)} I(h_m(\mathbf{x}) \neq t_n)}{\sum_{n=1}^N w_n^{(m)}}$$
 - Calculate a weighting coefficient for $h_m(\mathbf{x})$:

$$\alpha_m = \ln \left\{ \frac{1 - \epsilon_m}{\epsilon_m} \right\}$$
 - Update the weighting coefficients:

$$w_n^{(m+1)} = w_n^{(m)} \exp \{ \alpha_m I(h_m(\mathbf{x}_n) \neq t_n) \}$$

21

RWTH AACHEN UNIVERSITY

AdaBoost – Analysis

- Result of this derivation
 - We now know that AdaBoost minimizes an exponential error function in a sequential fashion.
 - This allows us to analyze AdaBoost's behavior in more detail.
 - In particular, we can see how robust it is to outlier data points.

22

RWTH AACHEN UNIVERSITY

Topics of This Lecture

- AdaBoost
 - Algorithm
 - Analysis
 - Extensions
- Analysis
 - Comparing Error Functions
- Applications
 - AdaBoost for face detection
- Decision Trees
 - CART
 - Impurity measures, Stopping criterion, Pruning
 - Extensions, Issues
 - Historical development: ID3, C4.5

23

RWTH AACHEN UNIVERSITY

Recap: Error Functions

$t_n \in \{-1, 1\}$

Ideal misclassification error

- Ideal misclassification error function (black)
 - This is what we want to approximate,
 - Unfortunately, it is not differentiable.
 - The gradient is zero for misclassified points.

⇒ We cannot minimize it by gradient descent.

24
Image source: Bishop, 2006

RWTH AACHEN UNIVERSITY

Recap: Error Functions

$t_n \in \{-1, 1\}$

Ideal misclassification error
Squared error

- Squared error used in Least-Squares Classification
 - Very popular, leads to closed-form solutions.
 - However, sensitive to outliers due to squared penalty.
 - Penalizes "too correct" data points

⇒ Generally does not lead to good classifiers.

25
Image source: Bishop, 2006

Machine Learning Winter '17

RWTH AACHEN UNIVERSITY

Recap: Error Functions

Robust to outliers!

Not differentiable!

Favors sparse solutions!

$E(z_n)$
 Ideal misclassification error
 Squared error
 Hinge error

$z_n = t_n y(x_n)$

- “Hinge error” used in SVMs
 - Zero error for points outside the margin ($z_n > 1$) \Rightarrow sparsity
 - Linear penalty for misclassified points ($z_n < 1$) \Rightarrow robustness
 - Not differentiable around $z_n = 1 \Rightarrow$ Cannot be optimized directly.

B. Leibe 26
Image source: Bishop, 2006

Machine Learning Winter '17

RWTH AACHEN UNIVERSITY

Discussion: AdaBoost Error Function

Robust to outliers!

Not differentiable!

$E(z_n)$
 Ideal misclassification error
 Squared error
 Hinge error
 Exponential error

$z_n = t_n y(x_n)$

- Exponential error used in AdaBoost
 - Continuous approximation to ideal misclassification function.
 - Sequential minimization leads to simple AdaBoost scheme.
 - Properties?

B. Leibe 27
Image source: Bishop, 2006

Machine Learning Winter '17

RWTH AACHEN UNIVERSITY

Discussion: AdaBoost Error Function

Sensitive to outliers!

$E(z_n)$
 Ideal misclassification error
 Squared error
 Hinge error
 Exponential error

$z_n = t_n y(x_n)$

- Exponential error used in AdaBoost
 - No penalty for too correct data points, fast convergence.
 - Disadvantage: exponential penalty for large negative values!
 \Rightarrow Less robust to outliers or misclassified data points!

B. Leibe 28
Image source: Bishop, 2006

Machine Learning Winter '17

RWTH AACHEN UNIVERSITY

Discussion: Other Possible Error Functions

$E = -\sum \{t_n \ln y_n + (1-t_n) \ln(1-y_n)\}$

$E(z_n)$
 Ideal misclassification error
 Squared error
 Hinge error
 Exponential error
 Cross-entropy error

$z_n = t_n y(x_n)$

- “Cross-entropy error” used in Logistic Regression
 - Similar to exponential error for $z > 0$.
 - Only grows linearly with large negative values of z .
 - \Rightarrow Make AdaBoost more robust by switching to this error function.
 - \Rightarrow “GentleBoost”

B. Leibe 29
Image source: Bishop, 2006

Machine Learning Winter '17

RWTH AACHEN UNIVERSITY

Summary: AdaBoost

- Properties
 - Simple combination of multiple classifiers.
 - Easy to implement.
 - Can be used with many different types of classifiers.
 - None of them needs to be too good on its own.
 - In fact, they only have to be slightly better than chance.
 - Commonly used in many areas.
 - Empirically good generalization capabilities.
- Limitations
 - Original AdaBoost sensitive to mislabeled training data points.
 - Because of exponential error function.
 - Improvement by GentleBoost
 - Single-class classifier
 - Multiclass extensions available

B. Leibe 30

Machine Learning Winter '17

RWTH AACHEN UNIVERSITY

Topics of This Lecture


- AdaBoost
 - Algorithm
 - Analysis
 - Extensions
- Analysis
 - Comparing Error Functions
- Applications
 - AdaBoost for face detection
- Decision Trees
 - CART
 - Impurity measures, Stopping criterion, Pruning
 - Extensions, Issues
 - Historical development: ID3, C4.5

B. Leibe 31

RWTH AACHEN UNIVERSITY

Example Application: Face Detection

- Frontal faces are a good example of a class where global appearance models + a sliding window detection approach fit well:
 - Regular 2D structure
 - Center of face almost shaped like a "patch"/window



- Now we'll take AdaBoost and see how the Viola-Jones face detector works


Machine Learning Winter '17 32

Slide credit: Kristen Grauman B. Leibe

RWTH AACHEN UNIVERSITY

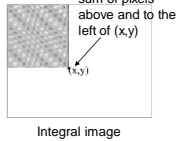
Feature extraction

"Rectangular" filters



Feature output is difference between adjacent regions

Efficiently computable with integral image: any sum can be computed in constant time



Value at (x,y) is sum of pixels above and to the left of (x,y)

Avoid scaling images → scale features directly for same cost

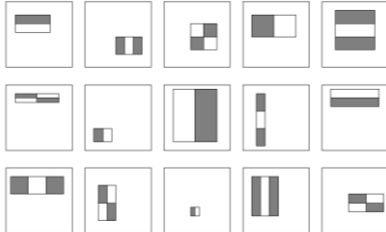
$$D = I(1) + I(4) - (I(2) + I(3))$$

Machine Learning Winter '17 33

Slide credit: Kristen Grauman B. Leibe [Viola & Jones, CVPR 2001]

RWTH AACHEN UNIVERSITY

Large Library of Filters



Considering all possible filter parameters: position, scale, and type:

180,000+ possible features associated with each 24 x 24 window

Use AdaBoost both to select the informative features and to form the classifier

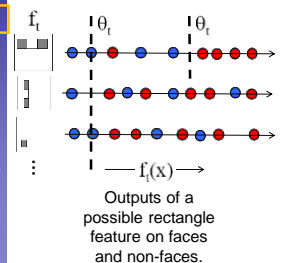
Machine Learning Winter '17 34

Slide credit: Kristen Grauman B. Leibe [Viola & Jones, CVPR 2001]

RWTH AACHEN UNIVERSITY

AdaBoost for Feature+Classifier Selection

- Want to select the single rectangle feature and threshold that best separates **positive** (faces) and **negative** (non-faces) training examples, in terms of *weighted error*.



Resulting weak classifier:

$$h_t(x) = \begin{cases} +1 & \text{if } f_t(x) > \theta_t \\ -1 & \text{otherwise} \end{cases}$$

For next round, reweight the examples according to errors, choose another filter/threshold combo.

Machine Learning Winter '17 35

Slide credit: Kristen Grauman B. Leibe [Viola & Jones, CVPR 2001]

RWTH AACHEN UNIVERSITY

AdaBoost for Efficient Feature Selection

- Image features = weak classifiers
- For each round of boosting:
 - Evaluate each rectangle filter on each example
 - Sort examples by filter values
 - Select best threshold for each filter (min error)
 - Sorted list can be quickly scanned for the optimal threshold
 - Select best filter/threshold combination
 - Weight on this features is a simple function of error rate
 - Reweight examples

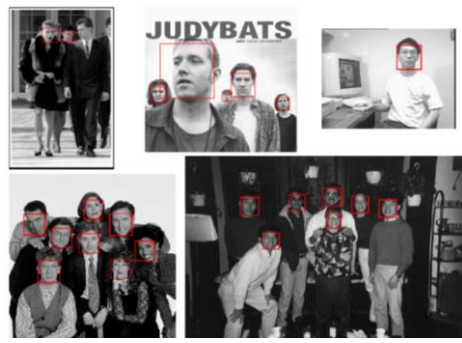
P. Viola, M. Jones, [Robust Real-Time Face Detection](#), IJCV, Vol. 57(2), 2004. (first version appeared at CVPR 2001)

Machine Learning Winter '17 36

Slide credit: Kristen Grauman B. Leibe

RWTH AACHEN UNIVERSITY

Viola-Jones Face Detector: Results



Machine Learning Winter '17 37

Slide credit: Kristen Grauman B. Leibe

RWTH AACHEN UNIVERSITY

Viola-Jones Face Detector: Results

Slide credit: Kristen Grauman B. Leibe


38

RWTH AACHEN UNIVERSITY

References and Further Reading

- More information on Classifier Combination and Boosting can be found in Chapters 14.1-14.3 of Bishop's book.

Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006


- A more in-depth discussion of the statistical interpretation of AdaBoost is available in the following paper:
 - J. Friedman, T. Hastie, R. Tibshirani, [Additive Logistic Regression: a Statistical View of Boosting](#), *The Annals of Statistics*, Vol. 38(2), pages 337-374, 2000.

B. Leibe

39

RWTH AACHEN UNIVERSITY

Topics of This Lecture

- AdaBoost
 - Algorithm
 - Analysis
 - Extensions
- Analysis
 - Comparing Error Functions
- Applications
 - AdaBoost for face detection
- Decision Trees
 - CART
 - Impurity measures, Stopping criterion, Pruning
 - Extensions, Issues
 - Historical development: ID3, C4.5


B. Leibe

40

RWTH AACHEN UNIVERSITY

Decision Trees

- Very old technique
 - Origin in the 60s, might seem outdated.
- But...
 - Can be used for problems with nominal data
 - E.g. attributes color \in {red, green, blue} or weather \in {sunny, rainy}.
 - Discrete values, no notion of similarity or even ordering.
 - Interpretable results
 - Learned trees can be written as sets of if-then rules.
 - Methods developed for handling missing feature values.
 - Successfully applied to broad range of tasks
 - E.g. Medical diagnosis
 - E.g. Credit risk assessment of loan applicants
 - Some interesting novel developments building on top of them...

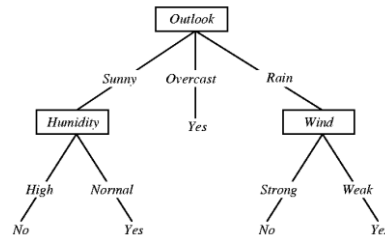


B. Leibe

41

RWTH AACHEN UNIVERSITY

Decision Trees



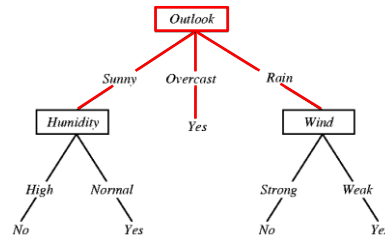
- Example:
 - "Classify Saturday mornings according to whether they're suitable for playing tennis."

B. Leibe Image source: T. Mitchell, 1997

42

RWTH AACHEN UNIVERSITY

Decision Trees



- Elements
 - Each node specifies a test for some attribute.
 - Each branch corresponds to a possible value of the attribute.

B. Leibe Image source: T. Mitchell, 1997

43

RWTH AACHEN UNIVERSITY

Decision Trees

- Assumption
 - Links must be mutually distinct and exhaustive
 - I.e. one and only one link will be followed at each step.

- Interpretability
 - Information in a tree can then be rendered as logical expressions.
 - In our example:

$$(Outlook = Sunny \wedge Humidity = Normal) \vee (Outlook = Overcast) \vee (Outlook = Rain \wedge Wind = Weak)$$

44
B. Leibe Image source: T. Mitchell, 1997

RWTH AACHEN UNIVERSITY

Training Decision Trees

- Finding the optimal decision tree is NP-hard...
- Common procedure: Greedy top-down growing
 - Start at the root node.
 - Progressively split the training data into smaller and smaller subsets.
 - In each step, pick the *best attribute* to split the data.
 - If the resulting subsets are pure (only one label) or if no further attribute can be found that splits them, terminate the tree.
 - Else, recursively apply the procedure to the subsets.
- CART framework
 - Classification And Regression Trees (Breiman et al. 1993)
 - Formalization of the different design choices.

45
B. Leibe

RWTH AACHEN UNIVERSITY

CART Framework

- Six general questions
 - Binary or multi-valued problem?
 - I.e. how many splits should there be at each node?
 - Which property should be tested at a node?
 - I.e. how to select the query attribute?
 - When should a node be declared a leaf?
 - I.e. when to stop growing the tree?
 - How can a grown tree be simplified or pruned?
 - Goal: reduce overfitting.
 - How to deal with impure nodes?
 - I.e. when the data itself is ambiguous.
 - How should missing attributes be handled?

46
B. Leibe

RWTH AACHEN UNIVERSITY

CART – 1. Number of Splits

- Each multi-valued tree can be converted into an equivalent binary tree:

⇒ Only consider binary trees here...

47
B. Leibe Image source: R.O. Duda, P.F. Hart, D.G. Stork, 2001

RWTH AACHEN UNIVERSITY

CART – 2. Picking a Good Splitting Feature

- Goal
 - Want a tree that is as simple/small as possible (Occam's razor).
 - But: Finding a minimal tree is an NP-hard optimization problem.
- Greedy top-down search
 - Efficient, but not guaranteed to find the smallest tree.
 - Seek a property T at each node s_j that makes the data in the child nodes as *pure* as possible.
 - For formal reasons more convenient to define *impurity* $i(s_j)$.
 - Several possible definitions explored.

48
B. Leibe

RWTH AACHEN UNIVERSITY

CART – Impurity Measures

- Misclassification impurity

$$i(s_j) = 1 - \max_k p(C_k | s_j)$$

"Fraction of the training patterns in category C_k that end up in node s_j ."

49
B. Leibe Image source: R.O. Duda, P.F. Hart, D.G. Stork, 2001

RWTH AACHEN UNIVERSITY

CART – Impurity Measures

- Entropy impurity

$$i(s_j) = - \sum_k p(C_k | s_j) \log_2 p(C_k | s_j)$$

“Reduction in entropy = gain in information.”

B. Leibe Image sources: R.O. Duda, P.F. Hart, D.G. Stork, 2001

RWTH AACHEN UNIVERSITY

CART – Impurity Measures

- Gini impurity (variance impurity)

$$i(s_j) = \sum_{k \neq l} p(C_k | s_j) p(C_l | s_j)$$

$$= \frac{1}{2} \left[1 - \sum_k p^2(C_k | s_j) \right]$$

“Expected error rate at node s_j , if the category label is selected randomly.”

B. Leibe Image sources: R.O. Duda, P.F. Hart, D.G. Stork, 2001

RWTH AACHEN UNIVERSITY

CART – Impurity Measures

- Which impurity measure should we choose?
 - Some problems with misclassification impurity.
 - Discontinuous derivative.
 - ⇒ Problems when searching over continuous parameter space.
 - Sometimes misclassification impurity does not decrease when Gini impurity would.
 - Both entropy impurity and Gini impurity perform well.
 - No big difference in terms of classifier performance.
 - In practice, stopping criterion and pruning method are often more important.

B. Leibe

RWTH AACHEN UNIVERSITY

CART – 2. Picking a Good Splitting Feature

- Application
 - Select the query that decreases impurity the most

$$\Delta i(s_j) = i(s_j) - P_L i(s_{j,L}) - (1 - P_L) i(s_{j,R})$$

P_L = fraction of points at left child node $s_{j,L}$
- Multway generalization (gain ratio impurity):
 - Maximize

$$\Delta i(s_j) = \frac{1}{Z} \left(i(s_j) - \sum_{m=1}^M P_m i(s_{j,m}) \right)$$
 - where the normalization factor ensures that large K are not inherently favored:

$$Z = - \sum_{m=1}^M P_m \log_2 P_m$$

B. Leibe

RWTH AACHEN UNIVERSITY

CART – Picking a Good Splitting Feature

- For efficiency, splits are often based on a single feature
 - “Monothetic decision trees”

- Evaluating candidate splits
 - Nominal attributes: exhaustive search over all possibilities.
 - Real-valued attributes: only need to consider changes in label.
 - Order all data points based on attribute x_i .
 - Only need to test candidate splits where $label(x_i) \neq label(x_{i+1})$.

B. Leibe

RWTH AACHEN UNIVERSITY

CART – 3. When to Stop Splitting


- Problem: Overfitting
 - Learning a tree that classifies the training data perfectly may not lead to the tree with the best generalization to unseen data.
 - Reasons
 - Noise or errors in the training data.
 - Poor decisions towards the leaves of the tree that are based on very little data.
- Typical behavior

Slide adapted from Raymond Mooney B. Leibe

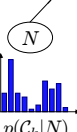
RWTH AACHEN UNIVERSITY

CART – Overfitting Prevention (Pruning)

- Two basic approaches for decision trees
 - Prepruning:** Stop growing tree as some point during top-down construction when there is no longer sufficient data to make reliable decisions.
 - Postpruning:** Grow the full tree, then remove subtrees that do not have sufficient evidence.
- Label leaf resulting from pruning with the majority class of the remaining data, or a class probability distribution.



$C_N = \arg \max_k p(C_k | N)$



$p(C_k | N)$

Slide adapted from Raymond Mooney. B. Leibe

56

RWTH AACHEN UNIVERSITY

Decision Trees – Computational Complexity

- Given
 - Data points $\{x_1, \dots, x_N\}$
 - Dimensionality D
- Complexity
 - Storage: $O(N)$
 - Test runtime: $O(\log N)$
 - Training runtime: $O(DN^2 \log N)$
 - Most expensive part.
 - Critical step: selecting the optimal splitting point.
 - Need to check D dimensions, for each need to sort N data points.

Machine Learning Winter '17 B. Leibe

66

RWTH AACHEN UNIVERSITY

Summary: Decision Trees

- Properties
 - Simple learning procedure, fast evaluation.
 - Can be applied to metric, nominal, or mixed data.
 - Often yield interpretable results.

Machine Learning Winter '17 B. Leibe

67

RWTH AACHEN UNIVERSITY

Summary: Decision Trees

- Limitations
 - Often produce noisy (bushy) or weak (stunted) classifiers.
 - Do not generalize too well.
 - Training data fragmentation:
 - As tree progresses, splits are selected based on less and less data.
 - Overtraining and undertraining:
 - Deep trees: fit the training data well, will not generalize well to new test data.
 - Shallow trees: not sufficiently refined.
 - Stability
 - Trees can be very sensitive to details of the training points.
 - If a single data point is only slightly shifted, a radically different tree may come out!
 - Result of discrete and greedy learning procedure.
 - Expensive learning step
 - Mostly due to costly selection of optimal split.

Machine Learning Winter '17 B. Leibe


68

RWTH AACHEN UNIVERSITY

References and Further Reading

- More information on Decision Trees can be found in Chapters 8.2-8.4 of Duda & Hart.

R.O. Duda, P.E. Hart, D.G. Stork
Pattern Classification
2nd Ed., Wiley-Interscience, 2000



Machine Learning Winter '17 B. Leibe

69