

RWTH AACHEN
UNIVERSITY

Machine Learning - Lecture 11

AdaBoost & Decision Trees

07.06.2016

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

Machine Learning, Summer '16

RWTH AACHEN
UNIVERSITY

Course Outline

- **Fundamentals (2 weeks)**
 - Bayes Decision Theory
 - Probability Density Estimation
- **Discriminative Approaches (5 weeks)**
 - Linear Discriminant Functions
 - Statistical Learning Theory & SVMs
 - Ensemble Methods & **Boosting**
 - Randomized Trees, Forests & Ferns
- **Generative Models (4 weeks)**
 - Bayesian Networks
 - Markov Random Fields

B. Leibe

2

RWTH AACHEN
UNIVERSITY

Recap: Stacking

- **Idea**
 - Learn L classifiers (based on the training data)
 - Find a meta-classifier that takes as input the output of the L first-level classifiers.

- **Example**
 - Learn L classifiers with leave-one-out.
 - Interpret the prediction of the L classifiers as L -dimensional feature vector.
 - Learn "level-2" classifier based on the examples generated this way.

B. Leibe

3

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

4

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

5

RWTH AACHEN
UNIVERSITY

Recap: AdaBoost - "Adaptive Boosting"

- **Main idea** [Freund & Schapire, 1996]
 - Instead of resampling, 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**
 - $h_m(x)$: "weak" or base classifier
 - Condition: <50% training error over any distribution
 - $H(x)$: "strong" or final classifier
- **AdaBoost:**
 - Construct a strong classifier as a thresholded linear combination of the weighted weak classifiers:

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

B. Leibe

6

RWTH AACHEN UNIVERSITY

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 $h_m(x)$ using the current weighting coefficients $\mathbf{W}^{(m)}$ by minimizing the weighted error function

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

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

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

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

How should we do this exactly?

8

RWTH AACHEN UNIVERSITY

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.

9

RWTH AACHEN UNIVERSITY

AdaBoost - Minimizing Exponential Error

- Exponential error function

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

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

10

RWTH AACHEN UNIVERSITY

AdaBoost - Minimizing Exponential Error

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

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

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

= const.

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

11

RWTH AACHEN UNIVERSITY

AdaBoost - Minimizing Exponential Error

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

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

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

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

12

RWTH AACHEN UNIVERSITY

AdaBoost - Minimizing Exponential Error

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

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

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

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

13

RWTH AACHEN UNIVERSITY

AdaBoost - Minimizing Exponential Error

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

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

⇒ 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)

⇒ We're on the right track. Let's continue...

14

RWTH AACHEN UNIVERSITY

AdaBoost - Minimizing Exponential Error

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

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

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

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

$$\epsilon_m = \frac{1}{e^{\alpha_m} + 1}$$

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

15

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

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

$$= \dots$$

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

⇒ Update for the weight coefficients.

16

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:

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

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

18

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.

19

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Recap: Error Functions

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

Not differentiable!

Ideal misclassification error

$E(z_n)$

$z_n = t_n y(\mathbf{x}_n)$

- **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.

20
Image source: Bishop, 2006

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Recap: Error Functions

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

Sensitive to outliers!

Penalizes "too correct" data points!

Ideal misclassification error

Squared error

$E(z_n)$

$z_n = t_n y(\mathbf{x}_n)$

- **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.

21
Image source: Bishop, 2006

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Recap: Error Functions

Robust to outliers!

Not differentiable!

Favors sparse solutions!

Ideal misclassification error

Squared error

Hinge error

$E(z_n)$

$z_n = t_n y(\mathbf{x}_n)$

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

22
Image source: Bishop, 2006

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Discussion: AdaBoost Error Function

Ideal misclassification error

Squared error

Hinge error

Exponential error

$E(z_n)$

$z_n = t_n y(\mathbf{x}_n)$

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

23
Image source: Bishop, 2006

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Discussion: AdaBoost Error Function

Sensitive to outliers!

Ideal misclassification error

Squared error

Hinge error

Exponential error

$E(z_n)$

$z_n = t_n y(\mathbf{x}_n)$

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

24
Image source: Bishop, 2006

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Discussion: Other Possible Error Functions

Ideal misclassification error

Squared error

Hinge error

Exponential error

Cross-entropy error

$E(z_n)$

$z_n = t_n y(\mathbf{x}_n)$

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

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

25
Image source: Bishop, 2006

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 misclassified training data points.
 - Because of exponential error function.
 - Improvement by GentleBoost
 - Single-class classifier
 - Multiclass extensions available

26

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Topics of This Lecture

- **Recap: 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

27

Machine Learning, Summer '16

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

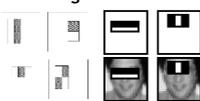
28

Machine Learning, Summer '16

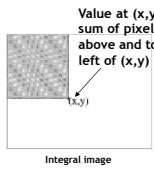
RWTH AACHEN UNIVERSITY

Feature extraction

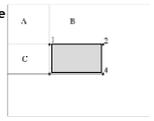
"Rectangular" filters



Feature output is difference between adjacent regions



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



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

Avoid scaling images → scale features directly for same cost

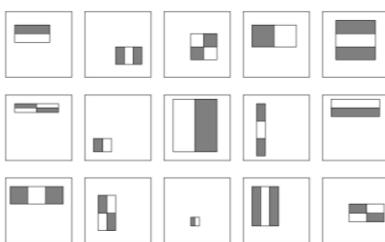
$$D = 1 + 4 - (2 + 3) = 1 + (1 + 3 + 3 + 3) - (1 + 3 + 1 + 3) = D$$

29

Machine Learning, Summer '16

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

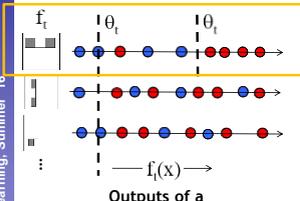
30

Machine Learning, Summer '16

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:

$$h_i(x) = \begin{cases} +1 & \text{if } f_i(x) > \theta_i \\ -1 & \text{otherwise} \end{cases}$$

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

31

Machine Learning, Summer '16

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)

32

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Viola-Jones Face Detector: Results

Slide credit: Kristen Grauman

33

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Viola-Jones Face Detector: Results

Slide credit: Kristen Grauman

34

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Viola-Jones Face Detector: Results

Slide credit: Kristen Grauman

35

Machine Learning, Summer '16

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.

36

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Topics of This Lecture

- Recap: 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

37

Machine Learning, Summer '16

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...

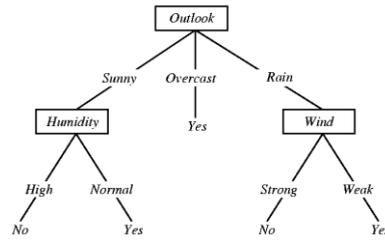


38

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Decision Trees



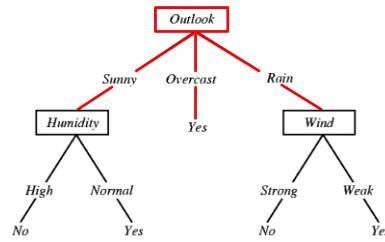
- Example:
 - “Classify Saturday mornings according to whether they’re suitable for playing tennis.”

39

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Decision Trees



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

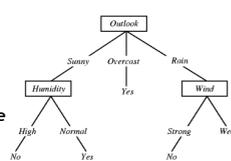
40

Machine Learning, Summer '16

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:
 - $(\text{Outlook} = \text{Sunny} \wedge \text{Humidity} = \text{Normal})$
 - $\vee (\text{Outlook} = \text{Overcast})$
 - $\vee (\text{Outlook} = \text{Rain} \wedge \text{Wind} = \text{Weak})$



41

Machine Learning, Summer '16

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.

42

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

CART Framework

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

43

Machine Learning, Summer '16

CART - 1. Number of Splits

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

⇒ Only consider binary trees here...

Machine Learning, Summer '16

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

44

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 N that makes the data in the child nodes as *pure* as possible.
 - For formal reasons more convenient to define *impurity* $i(N)$.
 - Several possible definitions explored.

Machine Learning, Summer '16

B. Leibe

45

CART - Impurity Measures

- Misclassification impurity

$$i(N) = 1 - \max_j p(C_j|N)$$

"Fraction of the training patterns in category C_j that end up in node N ."

Machine Learning, Summer '16

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

46

CART - Impurity Measures

- Entropy impurity

$$i(N) = - \sum_j p(C_j|N) \log_2 p(C_j|N)$$

"Reduction in entropy = gain in information."

Machine Learning, Summer '16

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

47

CART - Impurity Measures

- Gini impurity (variance impurity)

$$i(N) = \sum_{i \neq j} p(C_i|N)p(C_j|N)$$

$$= \frac{1}{2} [1 - \sum_j p^2(C_j|N)]$$

"Expected error rate at node N if the category label is selected randomly."

Machine Learning, Summer '16

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

48

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.

Machine Learning, Summer '16

B. Leibe

49

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

CART - 2. Picking a Good Splitting Feature

- Application
 - Select the query that decreases impurity the most

$$\Delta i(N) = i(N) - P_L i(N_L) - (1 - P_L) i(N_R)$$
- Multiway generalization (gain ratio impurity):
 - Maximize

$$\Delta i(s) = \frac{1}{Z} \left(i(N) - \sum_{k=1}^K P_k i(N_k) \right)$$
 - where the normalization factor ensures that large K are not inherently favored:

$$Z = - \sum_{k=1}^K P_k \log_2 P_k$$

B. Leibe 50

Machine Learning, Summer '16

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 51

Machine Learning, Summer '16

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 52

Machine Learning, Summer '16

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.

Slide adapted from Raymond Mooney. B. Leibe 53

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Decision Trees - Handling Missing Attributes

- During training
 - Calculate impurities at a node using only the attribute information present.
 - E.g. 3-dimensional data, one point is missing attribute x_3 .
 - Compute possible splits on x_1 using all N points.
 - Compute possible splits on x_2 using all N points.
 - Compute possible splits on x_3 using $N-1$ non-deficient points.
 - ⇒ Choose split which gives greatest reduction in impurity.
- During test
 - Cannot handle test patterns that are lacking the decision attribute!
 - ⇒ In addition to **primary split**, store an ordered set of **surrogate splits** that try to approximate the desired outcome based on different attributes.

B. Leibe 57

Machine Learning, Summer '16

RWTH AACHEN UNIVERSITY

Decision Trees - Feature Choice

- Best results if proper features are used

B. Leibe 58

RWTH AACHEN UNIVERSITY

Decision Trees - Feature Choice

Good tree

- Best results if proper features are used
 - Preprocessing to find important axes often pays off.

Machine Learning, Summer '16 59

RWTH AACHEN UNIVERSITY

Decision Trees - Non-Uniform Cost

- Incorporating category priors
 - Often desired to incorporate different priors for the categories.
 - Solution: weight samples to correct for the prior frequencies.
- Incorporating non-uniform loss
 - Create loss matrix λ_{ij}
 - Loss can easily be incorporated into Gini impurity

$$i(N) = \sum_{i,j} \lambda_{ij} p(C_i) p(C_j)$$

Machine Learning, Summer '16 60

RWTH AACHEN UNIVERSITY

Historical Development

- ID3 (Quinlan 1986)
 - One of the first widely used decision tree algorithms.
 - Intended to be used with nominal (unordered) variables
 - Real variables are first binned into discrete intervals.
 - General branching factor
 - Use gain ratio impurity based on entropy (information gain) criterion.
- Algorithm
 - Select attribute a that best classifies examples, assign it to root.
 - For each possible value v_i of a ,
 - Add new tree branch corresponding to test $a = v_i$.
 - If `example_list(vi)` is empty, add leaf node with most common label in `example_list(a)`.
 - Else, recursively call ID3 for the subtree with attributes $A \setminus a$.

Machine Learning, Summer '16 61

RWTH AACHEN UNIVERSITY

Historical Development

- C4.5 (Quinlan 1993)
 - Improved version with extended capabilities.
 - Ability to deal with real-valued variables.
 - Multiway splits are used with nominal data
 - Using gain ratio impurity based on entropy (information gain) criterion.
 - Heuristics for pruning based on statistical significance of splits.
 - Rule post-pruning
- Main difference to CART
 - Strategy for handling missing attributes.
 - When missing feature is queried, C4.5 follows all B possible answers.
 - Decision is made based on all B possible outcomes, weighted by decision probabilities at node N .

Machine Learning, Summer '16 62

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. $O(DN \log N)$

Machine Learning, Summer '16 63

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, Summer '16 64

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.

65

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



B. Leibe

66