Tree^2

Tree Patterns for Tree Decisions

Björn Bringmann Albrecht Zimmermann

Machine Learning Lab
Albert-Ludwigs-University Freiburg

Oberseminar Machine Learning

Outline

Outline

Outline

Outline

Outline

The Data

- Structured Data - Graphs, Trees, and Sequences
- Possibly described by additional non-structural features
- Describing some real (or not so real) life entities: Social Networks, Molecules, XML-structured Documents, Alarm sequences
- Some ”class” information: terrorist/SUV owner, mutagenicity, students surfing the WWW, true alarm/false alarm

The Task

- Classification: based on the structural information, predict class/concept an instance belongs to
- How to do this: extract structural pecularities, use them in classifier
- Why: to do viral marketing, find new vaccines, classify documents automatically, do intrusion detection

Outline

MolFea [De Raedt *et al.*]

- Molecules represented as linear sequences
- Mined subsequences frequent in actives, infrequent in inactives - significance based on (in-)frequency
- Can be used as features for ML algorithm (e.g. SVM)
- Decision has to be made on min-,max-frequency

FREQUENT SMILES [Karwath, Bringmann]

- Molecules represented as SMILES-parse-trees
- Mined subtrees with similar constraints as MOLFEA
- Used as features for SVM

XRules [Zaki *et al.*]

- XML data in k classes
- For each class, mine all subtrees having higher frequency than $s_i, i \in \{1, \ldots, k\}$ - frequency for significance
- Treat each subtree as rule predicting class it was mined from
- Prune rules with strength (confidence) less than δ_{min} - strength (confidence) for quality assessment
- For unseen instance, combine all matching rules
- Decision has to be made on k min-frequencies, min-strength

DT-GBI [Motoda *et al.*]

- Graph structured DNA data
- Directly build decision tree, inner nodes correspond to test of inclusion of certain subgraph
- Pre- and post-pruning options

DT-GBI - cont.

- How to find the subgraphs for tests?
- In each node:
  1. Assess quality of current candidates via *Information Gain* or such
  2. Order by frequency, quality
  3. Take k **most frequent** - frequency for significance assessment
  4. Specialize those
  5. Return to 1), unless specified maximum number of specializations has been performed or min-frequency violated
- Decisions have to be made on: beam size, number of max specializations/min-frequency

Outline

Choice of Representation

- We choose trees. Why?
- More information than flat representation (e.g. itemsets)
- Easier to deal with than graphs
- We have a treeminer

Choice of Classifier

- Integrated classifier: use the mined substructures directly. Why?
- Firstly, there is a understanding-gap if substructures get combined in some way by a classifier
- Secondly, since we use correlation measure to get high discriminative effect, features determine a lot

Choice of Quality Measure

- Correlation Measure (χ^2, *Information Gain*) - measuring correspondence of concept and pattern. Why?
- Confidence has a problem: empty rule, predicting majority in 2-class setting $\Rightarrow$ high confidence
- Also: what is ”good” confidence?

Choice of Significance Measure

- Cut-off value for quality measure. Why?
- Theoretical backing: e.g. significance values for χ^2-distribution
- Frequency difficult: step-function, how do you pick ”good” minimum frequency?

Outline

Tree^2(Meta)

FindSplit($\mathcal{D}$, σ, τ_{co})
$\mathcal{D}$ - dataset, σ - correlation measure, τ_{co} - user-specified cut-off value

- Find pattern p with highest discriminative power on $\mathcal{D}$, according to σ, with $\sigma(p) \geq \tau_{co}$
- Add node to DT including p
- Split $\mathcal{D}$ in $\mathcal{D}_+ = \{T \in \mathcal{D} | p \sqsubseteq T\}$, $\mathcal{D}_- = \{T \in \mathcal{D} | p \not\sqsubseteq T\}$
- FindSplit($\mathcal{D}_+$, σ, τ_{co})
- FindSplit($\mathcal{D}_-$, σ, τ_{co})
- Return DT

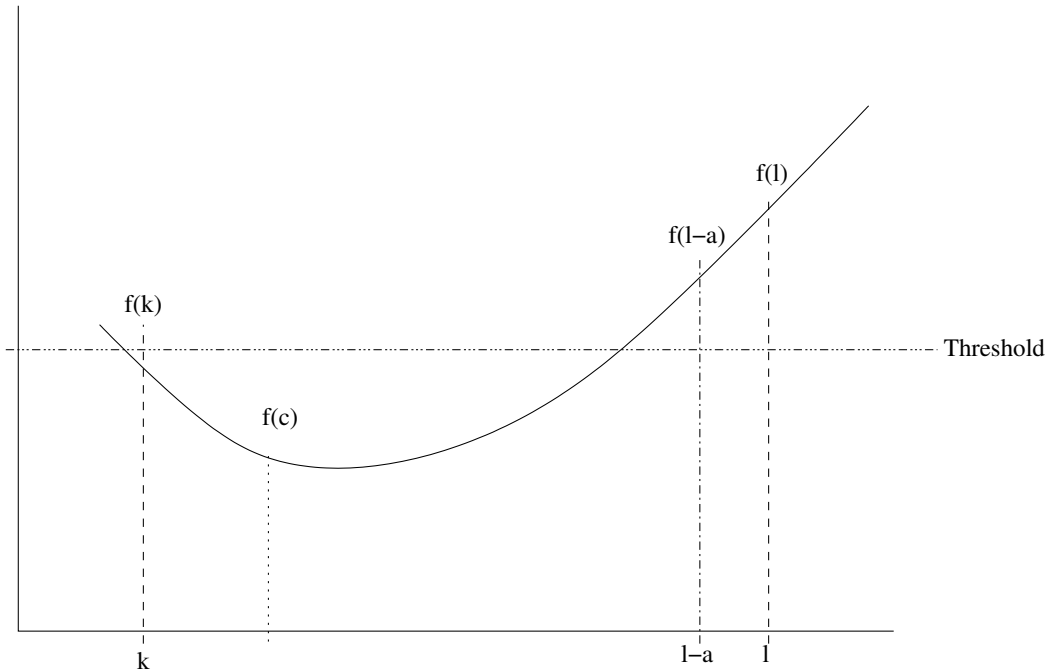
Tree^2 - cont.

- How to find p
- In each node:
  1. Start from most general pattern, $p_{best} = \emptyset$, $\tau = \tau_{co}$
  2. Create 1-step-specializations p', score with σ, calculate upper bound for each p'
  3. $p_{best} = \arg\max\{\sigma(p'), \sigma(p_{best})\}$
  4. $\tau = \max\{\tau, \tau_{co}\}$
  5. Prune all p' whose upper bound $< \tau$
  6. Goto 2)
  7. Return p_{best}, if $\sigma(p_{best}) \geq \tau_{co}$
- Upper bound calculation necessary, since σ not monotone

Outline

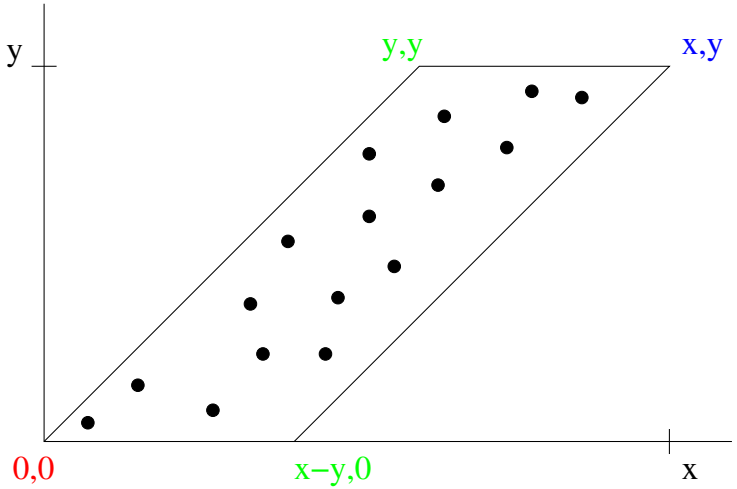
Convexity

- σ convex $\Rightarrow$ extreme values at convex borders



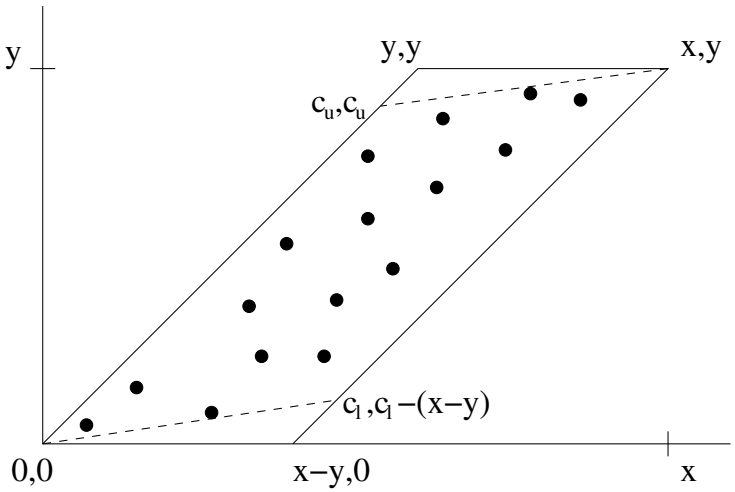
Convexity in 2D

- Basically the same, but now it's polygon vertices



Upper Bound on χ^2

- χ^2 only reliable if expected count ≥ 5
- Tighten convex hull
- Tightens upper bound as well



Comparison to XRules

- XML logs of surfers' sessions
- Mining with *chi*^{2} with 90%,95%,99% significance value as cut-off
- Also *Information Gain*, harder to define cut-off value
- Comparison to Zaki's published results
- Result: similar accuracy (slightly worse), smaller model (54 inner node *vs* 29000 rules)
- Trees get smaller with higher cut-off, accuracy best for 95%
- *Information Gain*: bigger trees (factor of 2), same accuracy

Comparison to Base-line Approach

- Mine 100 best subtrees according to σ
- Fed into *C4.5*
- Results: about the same accuracy, bigger trees
- Same kind of degenerate trees that we produce

Conclusion

- It works
- Trees are small, more understandable than base-line or XRules
- The horse is bit deader

Future Work

- (Really) Short Term: experiments with synthetic dataset, better distribution
- Re-do mutagenesis experiments
- Long Term: post-pruning, maybe other measures (depending on theory)

The End

Enjoy the Weekend