Structured Output Prediction on Data Streams

Sašo Džeroski
Jozef Stefan Institute, Ljubljana, Slovenia

MAESTRA
LEARNING FROM MASSIVE, INCOMPLETELY ANNOTATED, AND STRUCTURED DATA

Talk outline

- Predictive Modeling on Data Streams
- Structured Output Prediction
  - Multi-target regression
  - Multi-label classification
- Structured output prediction with predictive clustering
- SOP on data streams
  - MTR on data streams
  - MLC on data streams
- Further work

Predictive modeling: Classification and regression

	Descriptive space				**Target space**
Example 1	1	TRUE	0.49	0.69	Yes
Example 2	2	FALSE	0.08	0.07	Yes
Example 3	1	FALSE	0.08	0.07	No
Example 4	2	TRUE	0.49	0.69	Yes
Example 5	3	TRUE	0.49	0.69	No
Example 6	4	FALSE	0.08	0.07	Yes
…	…				…

	Descriptive space				**Target space**
Example 1	1	TRUE	0.49	0.69	0.84
Example 2	2	FALSE	0.08	0.07	0.75
Example 3	1	FALSE	0.08	0.07	0.11
Example 4	2	TRUE	0.49	0.69	0.52
Example 5	3	TRUE	0.49	0.69	0.35
Example 6	4	FALSE	0.08	0.07	0.78
…	…				…

Predictive Modeling on Data Streams

Predictive modeling from big data

- Large number of columns (high dimensionality)
- Large number of rows (massive data)
- Streaming rows (data streams)
  - All of the data are not available for access simultaneously
  - Data instances arrive at **high velocities**, in a **specific order** and their number is **potentially arbitrarily large**
  - The **underlying concept** (distribution) governing the data **can change (concept drift)**
  - The high velocity demands **fast processing**
  - The large and potentially infinite number of examples demands **economical management of available memory**

Data streams: Regression

	Descriptive space				Target space
…	…				…
Example n+5	**1**	**TRUE**	**0.49**	**0.69**	**0.45**
Example n+1	4	FALSE	0.08	0.07	0.12
Example n+2	6	FALSE	0.08	0.07	1.54
Example n+3	8	TRUE	0.00	1.00	3.12
Example n+4	6	TRUE	0.00	0.00	0.05
…	…				…

Structured-output prediction

Predictive modeling from complex data

- Multi-target prediction
  - Classification
  - Regression
  - Mixed
- Multi-label classification
  - Hierarchical multi-label classification
- Predicting (short) time series

Multi-target prediction

- Classification

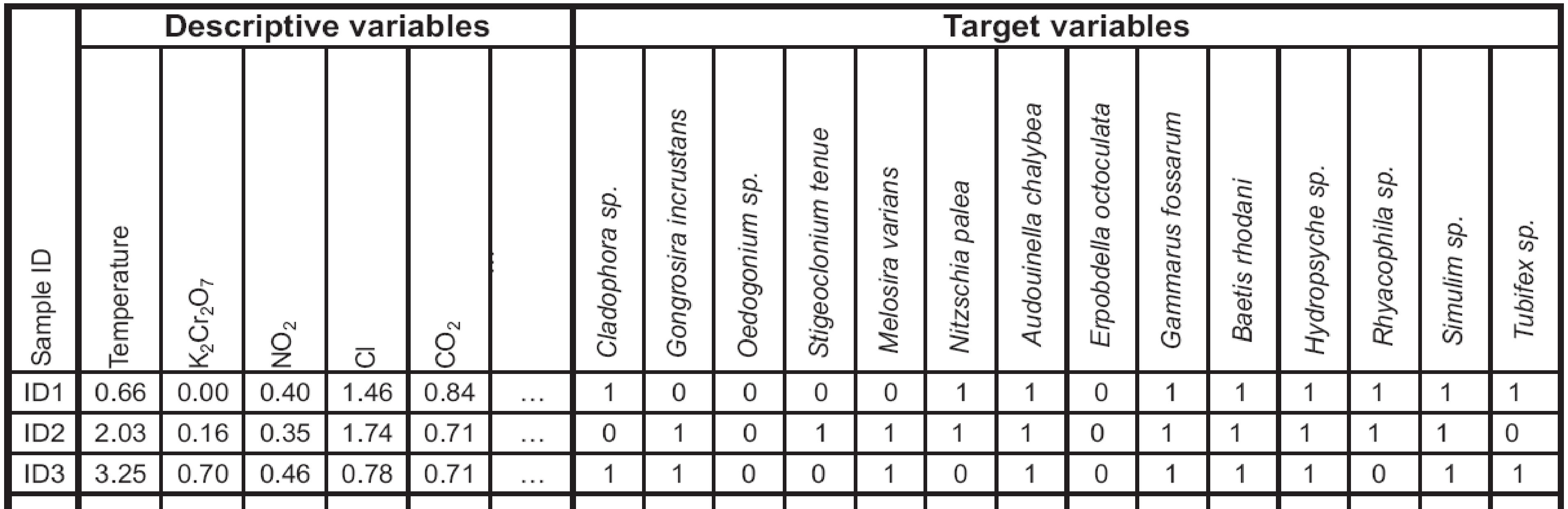
	Descriptive space				Target space		
Example 1	1	TRUE	0.49	0.69	Yes	Blue	Rain
Example 2	2	FALSE	0.08	0.07	Yes	Green	Sun
Example 3	1	FALSE	0.08	0.07	Yes	Blue	Cloudy
Example 4	2	TRUE	0.49	0.69	Yes	Green	Sun
Example 5	3	TRUE	0.49	0.69	No	Blue	Sun
Example 6	4	FALSE	0.08	0.07	Yes	Red	Cloudy
…	…				…	…	…

- Regression

	Descriptive space				Target space		
Example 1	1	TRUE	0.49	0.69	0.68	0.60	3.91
Example 2	2	FALSE	0.08	0.07	0.56	0.99	7.59
Example 3	1	FALSE	0.08	0.07	0.10	1.69	7.57
Example 4	2	TRUE	0.49	0.69	0.08	0.77	8.86
Example 5	3	TRUE	0.49	0.69	0.11	3.51	2.50
Example 6	4	FALSE	0.08	0.07	0.43	2.10	8.09
…	…				…	…	…

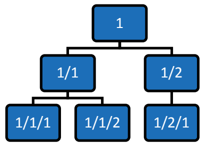
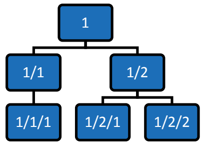
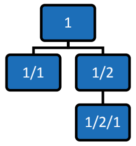
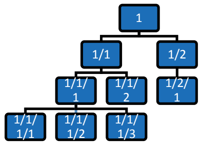
Multi-Label Classification

- Learning models that simultaneously predict several binary target variables
- Input: A vector of descriptive variables
- Output: A vector of several binary targets



Sample ID	Descriptive variables						Target variables													
	Temperature	$K_2Cr_2O_7$	NO_2	Cl	CO_2		*Cladophora sp.*	*Gongrosira incrustans*	*Oedogonium sp.*	*Stigeoclonium tenue*	*Melosira varians*	*Nitzschia palea*	*Audouinella chalybea*	*Erpobdella octoculata*	*Gammarus fossarum*	*Baetis rhodani*	*Hydropsyche sp.*	*Rhyacophila sp.*	*Simulim sp.*	*Tubifex sp.*
ID1	0.66	0.00	0.40	1.46	0.84	...	1	0	0	0	0	1	1	0	1	1	1	1	1	1
ID2	2.03	0.16	0.35	1.74	0.71	...	0	1	0	1	1	1	1	0	1	1	1	1	1	0
ID3	3.25	0.70	0.46	0.78	0.71	...	1	1	0	0	1	0	1	0	1	1	1	0	1	1

Hierarchical multi-label classification

	Descriptive space				Target space
Example 1	1	TRUE	0.49	0.69	
Example 2	2	FALSE	0.08	0.07	
Example 3	1	FALSE	0.08	0.07	
Example 4	2	TRUE	0.49	0.69	
…	…				…

SOP on Data Streams: Multi-Target Regression

	Descriptive space				**Target space**		
…	…				…		
Example n+5 Example n	6 1	TRUE	0.45 0.46	0.61 0.69	0.56 0.68	0.00 0.09	7.59 3.51
Example n+1	4	FALSE	0.08	0.07	0.10	1.69	7.57
Example n+2	6	FALSE	0.08	0.07	0.08	0.77	8.86
Example n+3	8	TRUE	0.00	1.00	0.11	3.51	2.50
Example n+4	6	TRUE	0.00	0.00	0.43	2.10	8.09
…	…				…	…	…

SOP on Data Streams: The Masterplan

- First, ST regression on data streams
  - Regression and model trees with change detection
  - Option trees
  - Ensembles
- Then, MT extension of ST regression algorithms
  - Multi-target regression/model trees
  - Option trees
  - Ensembles
- Next, (multi-label) classification via MT regression
- Finally, extension to hierarchical MTR (and HMC)

Top-Down Induction of Decision Trees

To construct a tree T from a training set S:

- If **all the examples belong to the same class C**, construct a leaf labeled C

- Otherwise:
  - Select the best attribute A with values v1, …, vn, which **most reduces the impurity (variance) of the target**
  - Partition S into S1, …, Sn according to A
  - Recursively construct subtrees T1 to Tn for S1 to Sn
  - Result: a tree with root A and subtrees T1, …, Tn

Learning regression trees on DS

Key difference to batch learning:

Data points arrive continuously and are sorted to tree leaves, where they accumulate until a split is possible

(we have enough evidence that one split is better

than all the others)

- We rate potential splits on their **variance reduction**
- For each leaf we find the two best splits
- We monitor their relative quality until examples accumulate in a leaf and we can be confident that the best test is indeed better than the second best one

Splitting a leaf

- A potential split over attribute A with threshold t separates a sample S accumulated in a leaf into two subsamples S_L and S_M, such that $S_L = \{s \in S;\ s_A \leq t\}$ and $S_M = \{s \in S;\ s_A > t\} = S - S_L$.
- We rate potential splits based on their **variance reduction** (VR).

$$VR(A,t) = Var(S) - Var(S_L) - Var(S_M),$$

Var is the variance of the target variable in the sample.

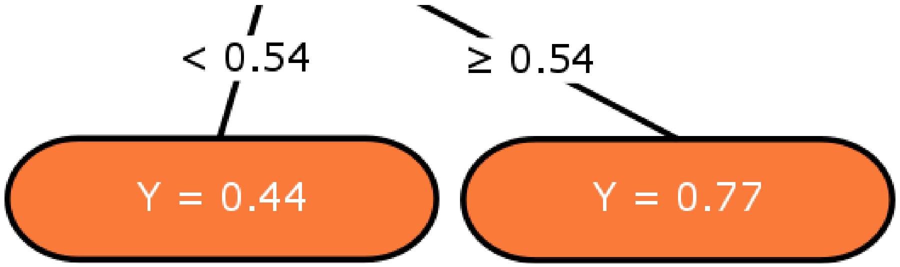
- The higher the variance reduction, the higher the purity of the subsamples and the more desirable the split.

Splitting a leaf

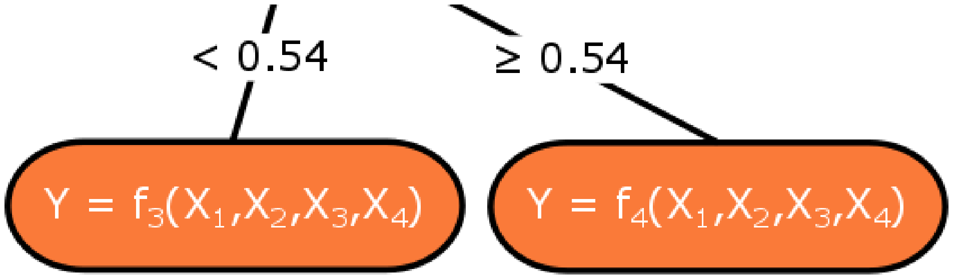
- For each leaf we find the two best splits .
- We monitor the ratio of the heuristic values of the best two splits, i.e., $\frac{VR(secondBest)}{VR(best)}$ and use the Hoeffding bound to probabilistically determine when the best split in fact outperforms the second best.
- The more examples accumulate in a leaf, the more certain we are. When a predefined threshold is passed, a leaf is split.
- Implemented in FIMT-DD

Regression vs. Model trees

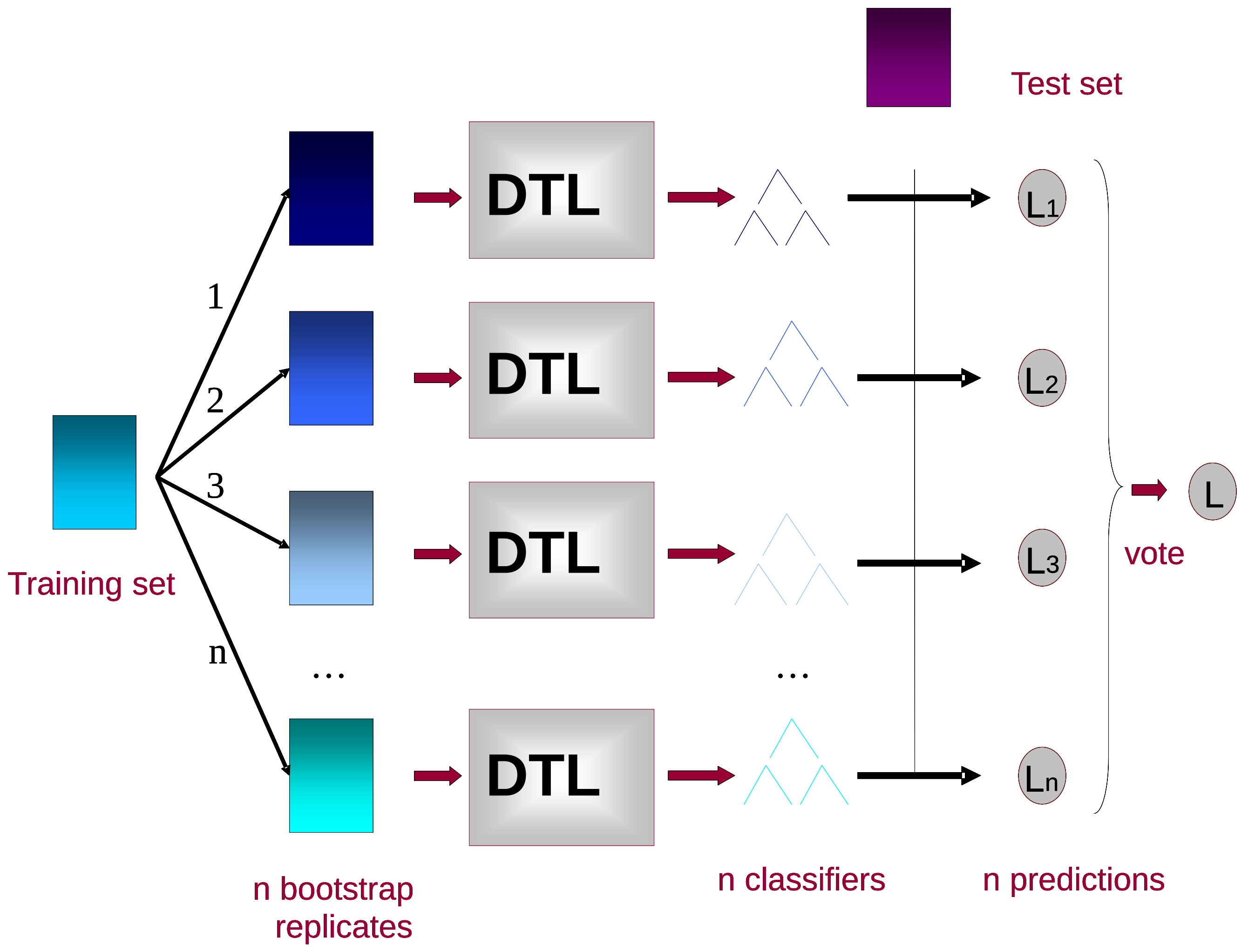
- Regression tree: each leaf holds a single value for the target.



- Model tree: each leaf contains a model, which computes the target value from the values of the input attributes.



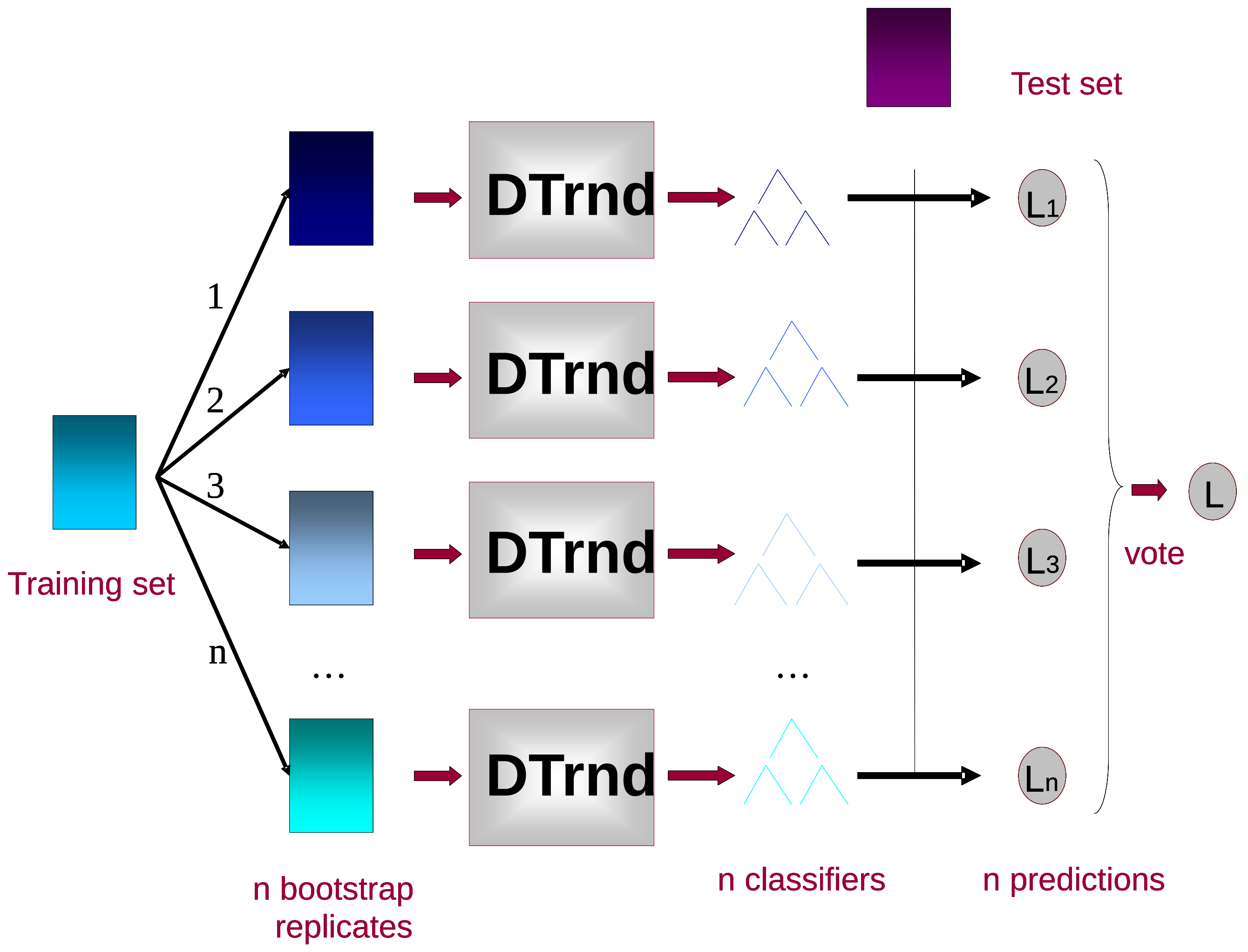
Ensembles of DT: Bagging



Ensembles on streams

- In the batch case, ensembles are trained on samples from the original dataset.
- In the streaming setting this is not feasible, since the dataset is not known in advance.
- There are adaptations of batch ensemble methods, such as **Online Bagging**, which consider how the algorithm would act if the size of the sample would approach inifinity.
- E.g., in online bagging each base model receives each data instance X number of times, where X is distributed according to the Poisson distribution, i.e., $X \sim Poisson(1)$.
- $P(\text{example is repeated } k \text{ times}) = \frac{1}{e \cdot k!}$

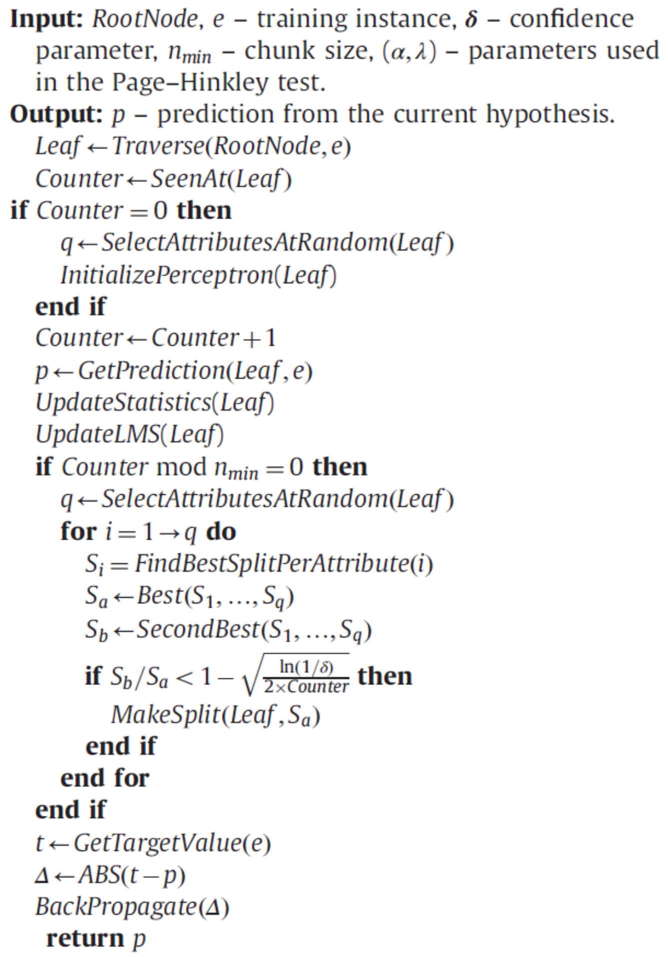
Ensembles of DT: Random Forests



ORF for ST regression on DS

- Online version of Random Forests
- Uses a modified base learner – R-FIMT-DD
  - R-FIMT-DD only considers a random subset of input features for splits in each new split node
  - $\sqrt{m}$ features are selected, where m is the total number of input features
  - Considerably reduces memory use
- Also online bagging w FIMT-DD

R-FIMT-DD algorithm, pseudocode



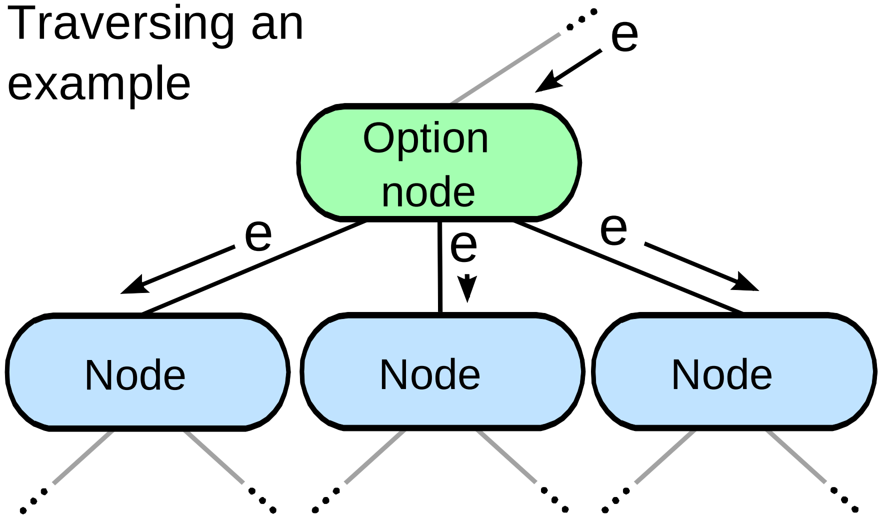
```
Input: $RootNode$, $e$ – training instance, $\delta$ – confidence
  parameter, $n_{min}$ – chunk size, $(\alpha, \lambda)$ – parameters used
  in the Page–Hinkley test.
Output: $p$ – prediction from the current hypothesis.
  $Leaf \leftarrow Traverse(RootNode, e)$
  $Counter \leftarrow SeenAt(Leaf)$
if $Counter = 0$ then
    $q \leftarrow SelectAttributesAtRandom(Leaf)$
    $InitializePerceptron(Leaf)$
  end if
  $Counter \leftarrow Counter + 1$
  $p \leftarrow GetPrediction(Leaf, e)$
  $UpdateStatistics(Leaf)$
  $UpdateLMS(Leaf)$
  if $Counter \bmod n_{min} = 0$ then
    $q \leftarrow SelectAttributesAtRandom(Leaf)$
    for $i = 1 \rightarrow q$ do
      $S_i = FindBestSplitPerAttribute(i)$
      $S_a \leftarrow Best(S_1, \ldots, S_q)$
      $S_b \leftarrow SecondBest(S_1, \ldots, S_q)$
      if $S_b / S_a < 1 - \sqrt{\frac{\ln(1/\delta)}{2 \times Counter}}$ then
        $MakeSplit(Leaf, S_a)$
      end if
    end for
  end if
  $t \leftarrow GetTargetValue(e)$
  $\Delta \leftarrow ABS(t - p)$
  $BackPropagate(\Delta)$
   return $p$
```

Option trees for regression

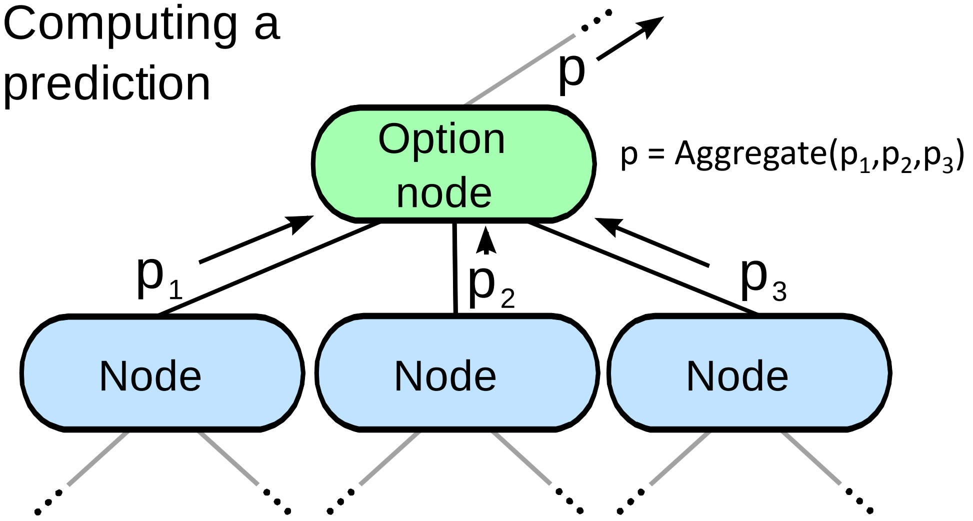
- Sometimes we cannot reliably determine which of the splits is the best.
- In such a case, an option tree selects multiple best ranked splits.
- When an option node is processing an example, the example is cloned and traversed through each of the splits. When the splits return the prediction, the option node aggregates them into a single prediction.
    - Select best prediction
    - Average predictions
- Option trees can be viewed as a compact representation of an ensemble (esp. in the latter case).

Option trees for regression

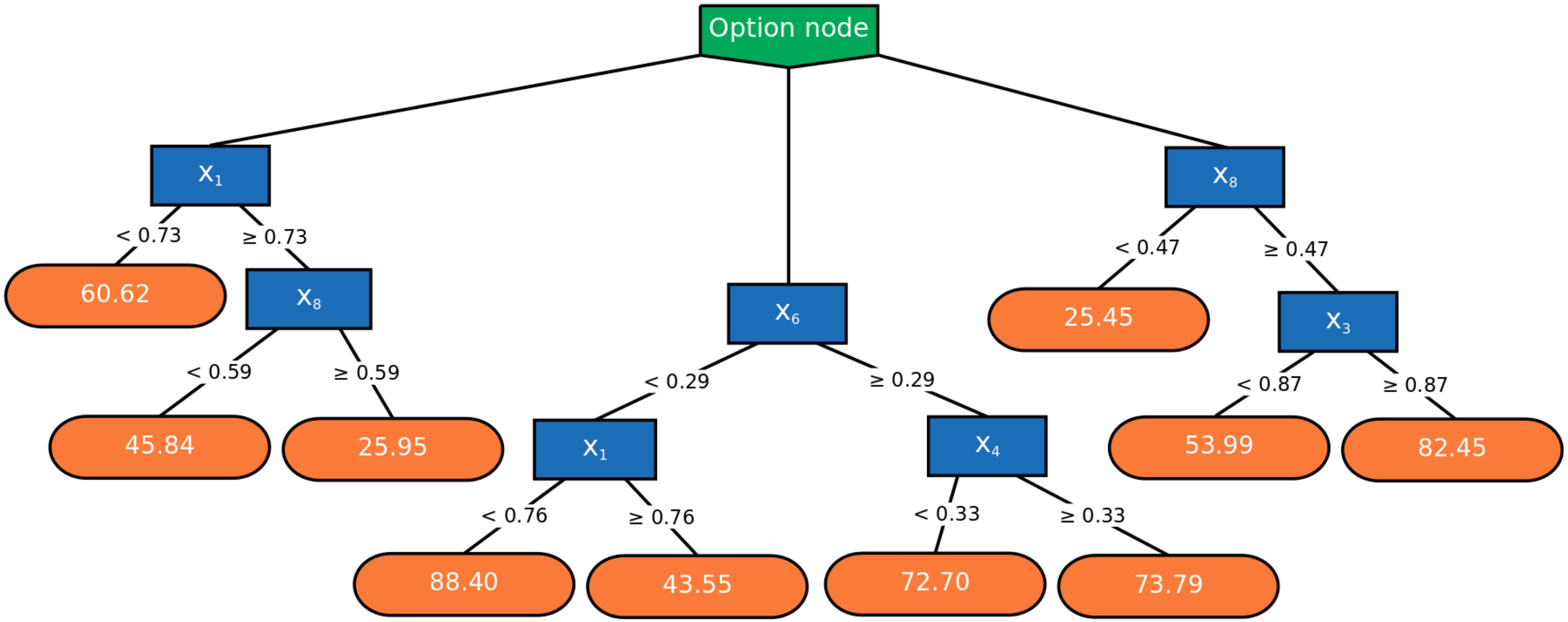
Traversing an example



Computing a prediction



Option tree for regression: Example



Top-down induction of PCTs

To construct a tree T from a training set S:

- If **the examples in S have low variance**,

  construct a leaf labeled *target(prototype(S))*
- Otherwise:
  - Select the best attribute A with values v1, …, vn, which **reduces the most the variance** (*measured according to a given distance function d*)
  - Partition S into S1, …, Sn according to A
  - Recursively construct subtrees T1 to Tn for S1 to Sn
  - Result: a tree with root A and subtrees T1, …, Tn

Distances/variances for SOP tasks

- The algorithm

```
procedure BestTest(E)
1: $(t^*, h^*, \mathcal{P}^*) = (none, 0, \emptyset)$
2: for each possible test $t$ do
3:     $\mathcal{P}$ = partition induced by $t$ on $E$
4:     $h = Var(E) - \sum_{E_i \in \mathcal{P}} \frac{|E_i|}{|E|} Var(E_i)$
5:     if $(h > h^*) \wedge \text{Acceptable}(t, \mathcal{P})$ then
6:         $(t^*, h^*, \mathcal{P}^*) = (t, h, \mathcal{P})$
7: return $(t^*, h^*, \mathcal{P}^*)$
```

- Variance for MT regression

$$Var(E) = \sum_{i=1}^{T} Var(Y_i).$$

- Variance for MT classification

$$Var(E) = \sum_{i=1}^{T} Entropy(E, Y_i)$$

- Variance for HMLC

$$Var(E) = \frac{1}{|E|} \cdot \sum_{E_i \in E} d(L_i, \overline{L})^2 \qquad d(L_1, L_2) = \sqrt{\sum_{l=1}^{|L|} w(c_l) \cdot (L_{1,l} - L_{2,l})^2}$$

From ST- to MT-regression on DS

- When addressing the task of (hierarchical) multi-target regression, we use the corresponding **intra-cluster variance reduction heuristic** (ICVR) as in PCTs

$$ICVR = \mathrm{IVar(S)} - \mathrm{IVar(S_L)} - \mathrm{IVar(S_M)},$$

where

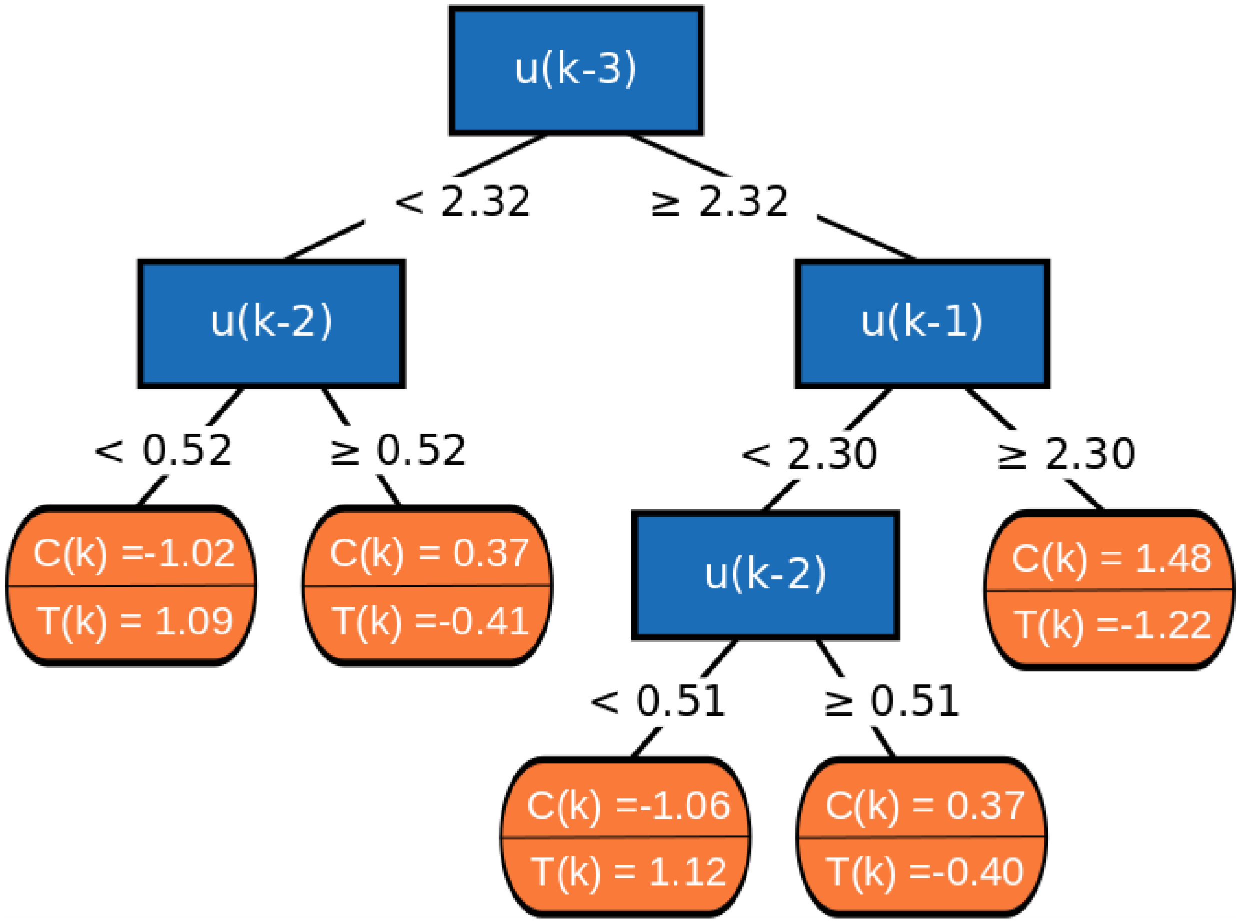
$$IVar(S) = \frac{1}{n}\sum_{i=1}^{n} Var(Y_i, S) \text{ and } Var(Y_i, S)$$

is the variance of the i-th target variable on sample S

- For HMTR, we use weights which are higher at higher levels of the hierarchy and decrease as we go downwards

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} \sqrt{w(c_i)\,(x_i - y_i)^2}, \quad w(c_i) = w_0^{\mathrm{level}(c_i)},\ w_0 < 1$$

A MT regression tree learnt on a DS



MLC via MTR

$$\begin{tikzcd}
 & \textbf{Target space} & & \textbf{Instance} \\
\textbf{MLC} & y \subseteq \mathcal{L} = \{\lambda_1, \ldots, \lambda_n\} \arrow[d] & & y = \{\lambda_1, \lambda_3, \lambda_4\} \arrow[d] \\
 & & \textit{transformation} & \\
\textbf{MTR} & y \in \mathbb{R}^n & & y = (1, 0, 1, 1, \ldots)
\end{tikzcd}$$

From a MLC to a MTR space… and back again.

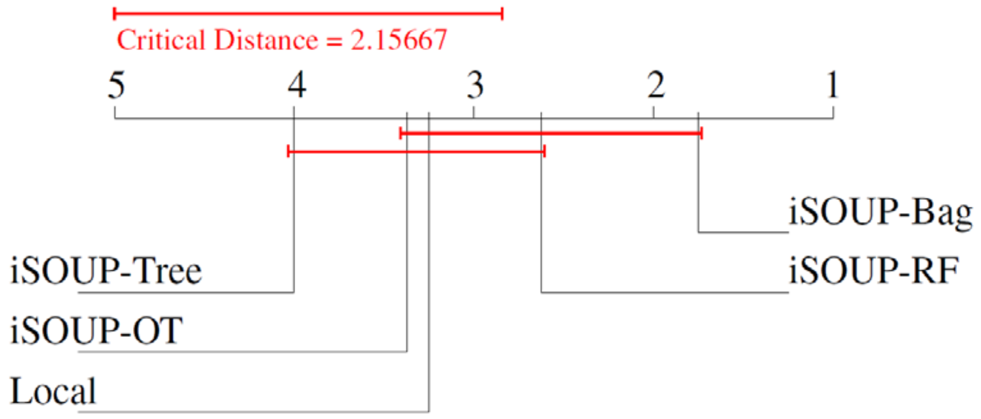
$$\begin{tikzcd}
 & \textbf{Target space} & & \textbf{Instance} \\
\textbf{MTR} & \hat{y} \in \mathbb{R}^n \arrow[d] & & \hat{y} = (0.98, 0.21, 0.59, 0.88, \ldots) \arrow[d] \\
 & & \textit{thresholding} & \\
\textbf{MLC} & \hat{y} \subseteq \mathcal{L} & & \hat{y} = \{\lambda_1, \lambda_3, \lambda_4\}
\end{tikzcd}$$

The implementation: iSOUPTrees

- Incremental Structured Output Prediction Trees - **iSOUPTrees**
- Started from FIMT-DD implementation of STRTs in VFML
- Implemented in the MOA framework
  - Increases usability and visibility
- Single-target regression algorithms & (H)MT extensions
  - Single-target regression/model trees
  - Option trees
  - Ensembles
- Support for nominal attributes (not present in FIMT)
  - Crucial – especially for MLC, as most MLC datasets consist of exclusively nominal attributes

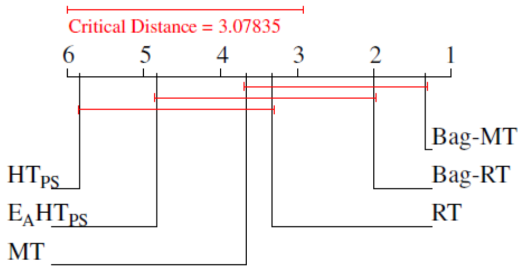
Exp. Eval.: MTR

- Bagging works best
- Random forests provide best trade-off between performance and resource consumption
- Results different from ST case, where option trees performed best

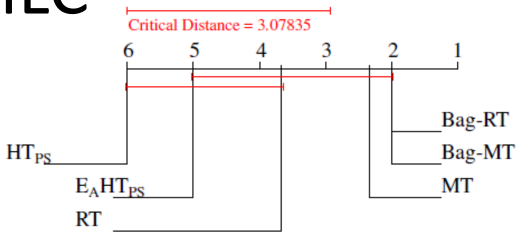


Exp. Eval.: MLC

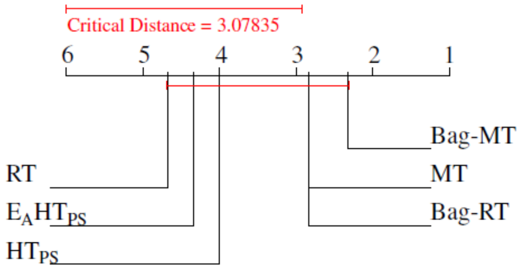
- Bagging of multi-target trees works best
- Followed closely by bagging of single-target trees
- Clearly better than the competition (Jesse and Albert)
- In terms of ranking-based measures, which are most relevant to MLC



(a) Ranking loss



(b) Logarithmic loss



(c) Average precision

Further work

- Option trees for SOP in the batch setting: paper at Discovery Science 2016 in Bari
- Hierarchical MTR: Evaluation (if you have a dataset, please let us know)
- Change detection in trees for SOP on DS
- Semi-supervised SOP on data streams

Acknowledgements and announcement

We acknowledge European Commission support through the grants

- MAESTRA: Learning from Massive, Incompletely annotated, and Structured Data, grant 612944
- HBP SGA1: The Human Brain Project, grant 720270
- LANDMARK: LAND Management: Assessment, Research, Knowledge base, grant 635201

As well as the Slovenian Research Agency through

- P2-0103 Knowledge technologies
- L2-7509 Structured output prediction …

And announce …

Thanks for coming to our Summer School Mining Big& Complex Data 4-8 SEP 2016, Ohrid

ECML PKDD 2017

SKOPJE, MACEDONIA

18-22 September 2017