Biomedical Decision Support System
academic year 2023/2024

Prof. Pietro Sala
Department of Computer Science
University of Verona
pietro.sala@univr.it

Project 3.a: *Timeseries Pattern Tree*

Context

A time-series ts is any sequence in $\mathbb{R}^*$. A *supervised dataset of timeseries* Ts is a multiset of pairs in $\mathbb{R}^* \times L$ where L is a finite set of class labels, i.e., $L = \{0, \dots, C-1\}$, that is the first C natural numbers.
A time-series pattern pts, a pattern from now on, is an element of $(\mathbb{R} \times \{+, -\})^*$. A time-series ts is said to contain a pattern pts, written $ts \models pts$ if there exists a subsequence $ts' \sqsubseteq ts$ such that $|ts'| = |pts|$ and for every $0 \le i < |ts'|$ we have $ts'[i] \ge pts[i][0]$ if $pts[i][1] = +$ and $ts'[i] \le pts[i][0]$ otherwise.
A time-series pattern rule, a rule from now on, is a pair $rpts = (pts, i)$ where pts is a pattern with $|pts| \ge 2$ and i is an index $0 < i < |pts|$. Its association rule may be graphically represented as:

$$pts[0:i] \to pts[i:]$$

A time-series pattern rule tree, a tree from now on, is a labelled rooted-tree $(T = (V, E = E_\perp \cup E_\top), r, \mathcal{V}, \mathcal{C})$ with root r where $E_\perp \cap E_\top$ and for every $v \in V$ we have $|\{v' : (v, v') \in E_x\}| \le 1$ for each $x \in \{\top, \perp\}$, since they are both unique we denote with $v_\top$ the unique successor (if any) of v in $E_\top$ and with $v_\perp$ the unique successor (if any) of v in $E_\perp$. Each node v is labeled with a pattern $rpts_v = \mathcal{V}(v)$ and a class label $\mathcal{C}_v = \mathcal{C}(v)$.

Assignment

Given a supervised dataset of timeseries Ts, a threshold $0 \le \epsilon \le 1$, and a mapping $\mathcal{B} : \mathbb{M} \to \mathbb{I}(\mathbb{R} \cup \{-\infty, +\infty\})$ from interesting measures for rules to an interval oin the reals a loss function $\mathcal{L} : (\mathcal{L} \to \mathbb{N}^2) \to \mathbb{R}$ implement a function *rpts-tree*$(Ts, \mathcal{M}, \mathcal{L})$ builds a tree T recursively as follows (function begins with $E_\top = E_\perp = \emptyset$):

1. v is a fresh node with $\mathcal{L}(r) = \emptyset$ and $\mathcal{C}(r) = \emptyset$;
2. let $Ts_v = Ts$;
3. let $\mathcal{C}(v) = \arg\max_{c \in L} \sum_{(ts,c) \in Ts_v} Ts(ts, c)$;
4. if $\{(ts, c') \in \mathcal{C}(v)\} = \emptyset$ then return v;
5. let $RPTS_v = \{rpts : \forall im \in \mathbb{M}(im(Ts, rpts) \in \mathcal{B}(im))\}$
6. let $rpts_v$ be the rule that satisfies
$$\arg\max_{RPTS_v} \mathcal{L}\left(\left\{c \mapsto \left(\sum_{ts \models rpts} Ts(ts, c), \sum_{ts \not\models rpts} Ts(ts', c)\right) : c \in L\right\}\right)$$
7. let $\mathcal{V}(v) = rpts_v$
8. let $Ts_{v_\top} = \{(ts, c) \in Ts : ts \models rpts_v\}$
9. let $Ts_{v_\perp} = \{(ts, c) \in Ts : ts \not\models rpts_v\}$
10. $E = \begin{array}{c}(E_\top \cup \{(v, rpts\text{-}tree(Ts_\top, \mathcal{M}, \mathcal{L}))\}) \\ \cup \\ (E_\perp \cup \{(v, rpts\text{-}tree(Ts_\perp, \mathcal{M}, \mathcal{L}))\})\end{array}$
11. return v.

Implement the function *rpts-tree* with the satandard parameters *max_height* and *min_samples*. As loss function Information Gain may be considered. For interesting measures at least support and confidence should be considered.

Note

Candidates for $RPTS_v$ (step 5) may be generated with the following approach for avoid computationally expensive search space exploration:

1. set I to $\emptyset$;
2. set r to $\emptyset$;
3. pick a random ts from Ts_v and a random $i \notin I$ such that $0 \le i < |ts|$, add i to I;
4. pick a random ts' from Ts_v;
5. add $(ts[i], ts'[i] >= ts[i], i)$ to r;
6. let $sorted(I) = i_1 < \ldots < i_k$ and pts be the pattern $[(ts[i_j]^*) : (ts[i_j], *, i_j) \in r, i_j insorted(I)]$;
7. if $|pts|$ is less than 2 then go to step 3;
8. else if $|pts|$ is greater than 2 and pts is supported;
9. if there exists i such that $pts[0:i] \to pts[i:]$ satisfy the interesting measures let i for which such a condition hold then $rpts = (pts, i)$;
10. if the previous condition does not hold return $rpts$.

Datasets

There are available dataset at [1].

Project 3.b: *Sequence Boosting*

Context

Given a dataset of $L = \{-1, +1\}$ labelled sequences, $Z \subseteq (A^+ \times L)$ on any finite alphabeth A

Assignment

proceed by implementing the following boosting classifier:

$$sgn(\alpha_1(h^1, l^1) + \ldots + \alpha_n(h^n, l^n))$$

where $h_i \in A^+$, $l \in L$ are sequences and α_i are the weights of the boosting scoring polynomial. The boosting algorithm is as follows:

1. set $t = 1$, $w_i^t = \frac{1}{|L|}$ for each $1 \le i \le |L|$;
2. find the best sequence h^t, l^t that minimizes the error rate:
$$h^t, l^t = \arg\min_{(h,l) \in A^+ \times L} \left(\begin{array}{c}\sum_{(x_i,y_i) \in Z, h \sqsubseteq x \wedge y \ne l} w_i^t \\ + \\ \sum_{(x_i,y_i) \in Z, h \not\sqsubseteq x \wedge y = l} w_i^t\end{array}\right)$$
3. set $\epsilon^t = \sum_{(x_i,y_i) \in Z, h_i \sqsubseteq x \wedge y \ne l_i} w_i^t + \sum_{(x_i,y_i) \in Z, h \not\sqsubseteq x \wedge y = l} w_i^t$;

4. set
$$\alpha^t = \frac{1}{2}\log\frac{1-\epsilon^t}{\epsilon^t}$$
where ϵ^t is the error rate of the classifier h^t;
5. for each i we have
$$w_i^{t+1} = \begin{cases} \frac{w_i^t}{2\sum\limits_{(h^t,l^t)\models(x_j,y_j)} w_j^t} & \text{if } (h^t,l^t)\models(x_i,y_i) \\ \frac{w_i^t}{2\sum\limits_{(h^t,l^t)\not\models(x_j,y_j)} w_j^t} & \text{otherwise} \end{cases}$$
where $(h^t,l^t) \models (x_j,y_j)$ is true if and only if either $h^t \sqsubseteq x_j \wedge l^t = y_j$ or $h^t \not\sqsubseteq x_j \wedge l^t \neq y_j$;
6. set $t = t+1$.
7. repeat until $H = sgn(\alpha_1(h^1,l^1)+\ldots+\alpha_n(h^n,l^n))$ classify correctly all the sequences in Z or the error of H on Z is less than a given threshold δ.

Datasets

Sequence Dataset are available at [3], from the paper [2].

References

[1] Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, Gustavo Batista, and Hexagon-ML. The ucr time series classification archive, October 2018. `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`.

[2] Zengyou He, Ziyao Wu, Guangyao Xu, Yan Liu, and Quan Zou. Decision tree for sequences. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):251–263, 2023.

[3] Ziyao Wu. Seqdt, April 2020. `https://github.com/ZiyaoWu/SeqDT/tree/master/data`.