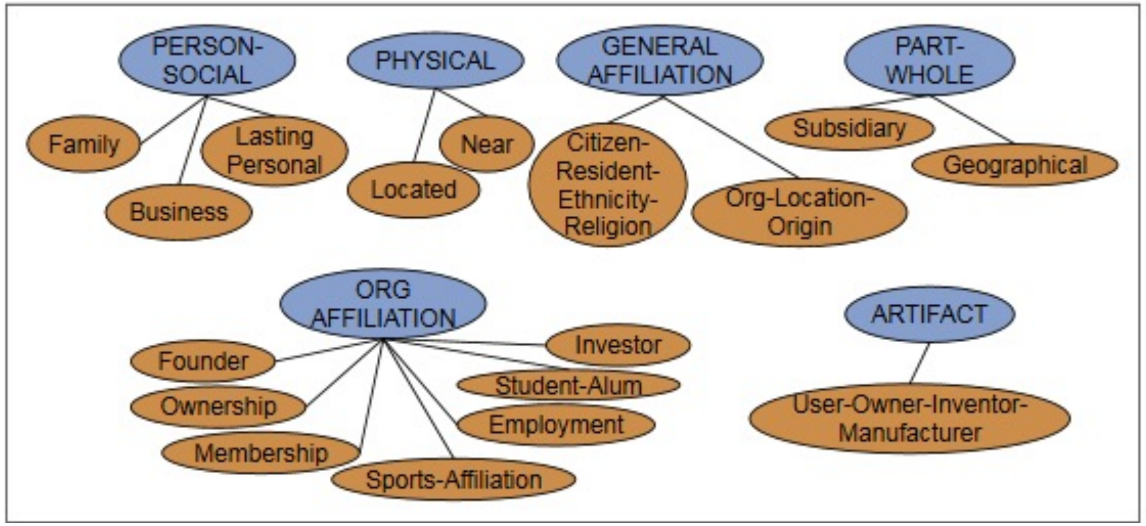
Machine Learning

Non structural hierarchical classification

Non-structured hierarchical classification (NSHC)

- Let us revisit a typical labelset appearing during relation extraction:

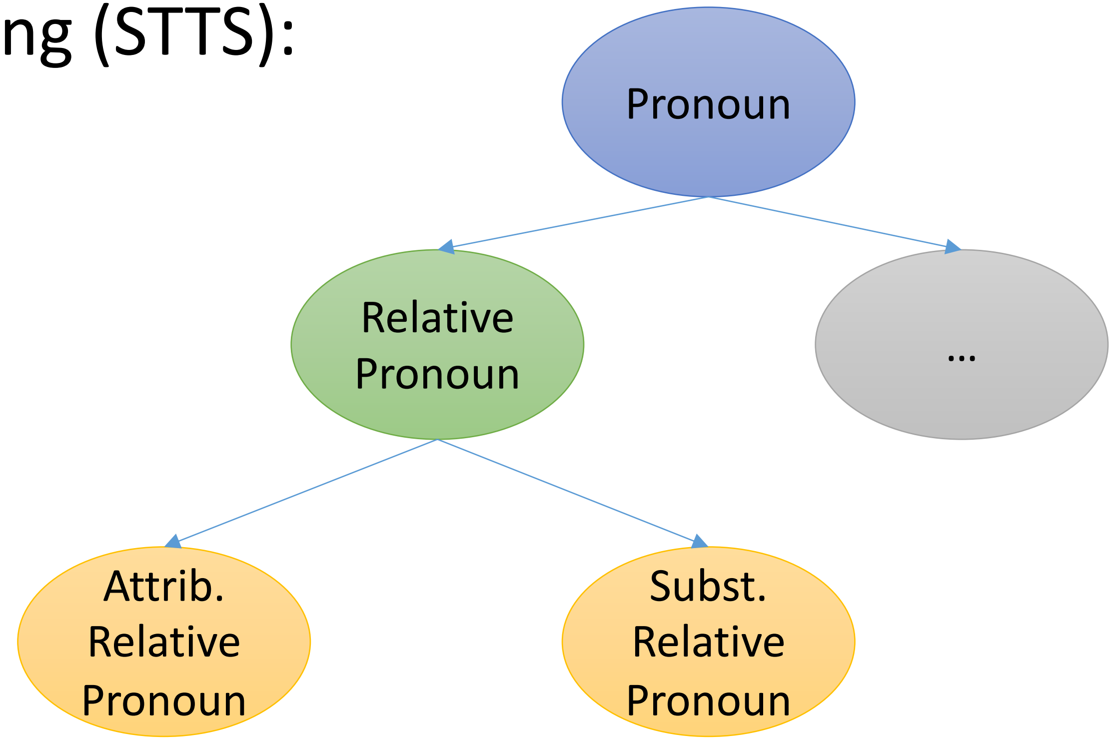


- If we include a <root> node, we got 3 layers of the hierarchy

➔ Let us make use of this information during our classification process!

Non-structured hierarchical classification (NSHC)

- Rethinking your current task, you might even find that your labels do also have some sort of hierarchy
- E.g. Part-of-Speech-tagging (STTS):



Non-structured hierarchical classification (NSHC)

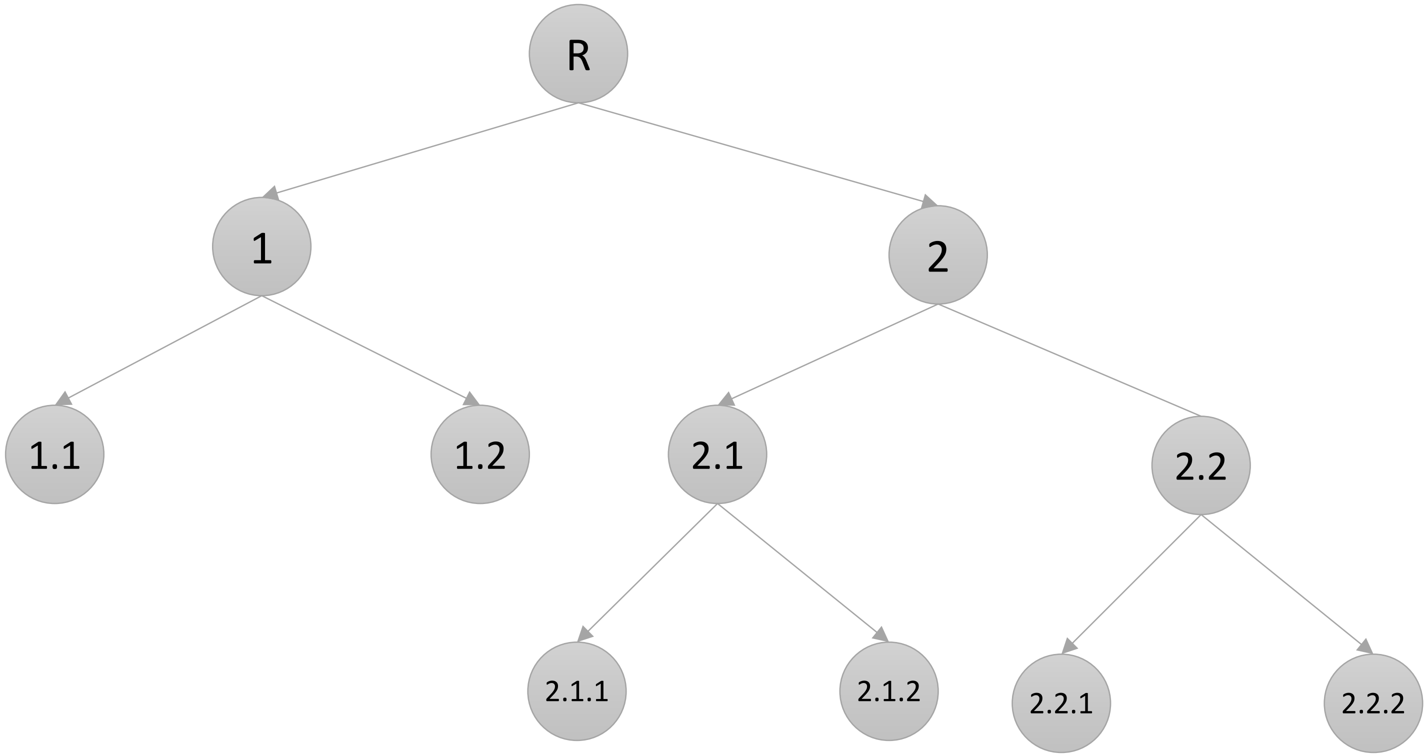
- We define our problem as a NSHC, if our labelset fulfills the following conditions:

The set of classes C is in a relation $\prec$ The relation $\prec$ is pronounced „IS-A“ relation and is ***asymmetric***, ***anti-reflexive*** und ***transitive***:

1. There is a single biggest element R („root“)
2. $\forall c_i, c_j \in C : \; if \; c_i \prec c_j \; then \; c_j \nprec c_i$ (asymmetric)
3. $\forall c_i \in C, c_i \nprec c_i$ (anti-reflexive)
4. $\forall c_i, c_j, c_k \in C, c_i \prec c_j \; and \; c_j \prec c_k \; implies \; c_i \prec c_k$(transitivity)

➔ And we will see, that our solution will work with any arbitrary classifier!

Running Example for NSHC



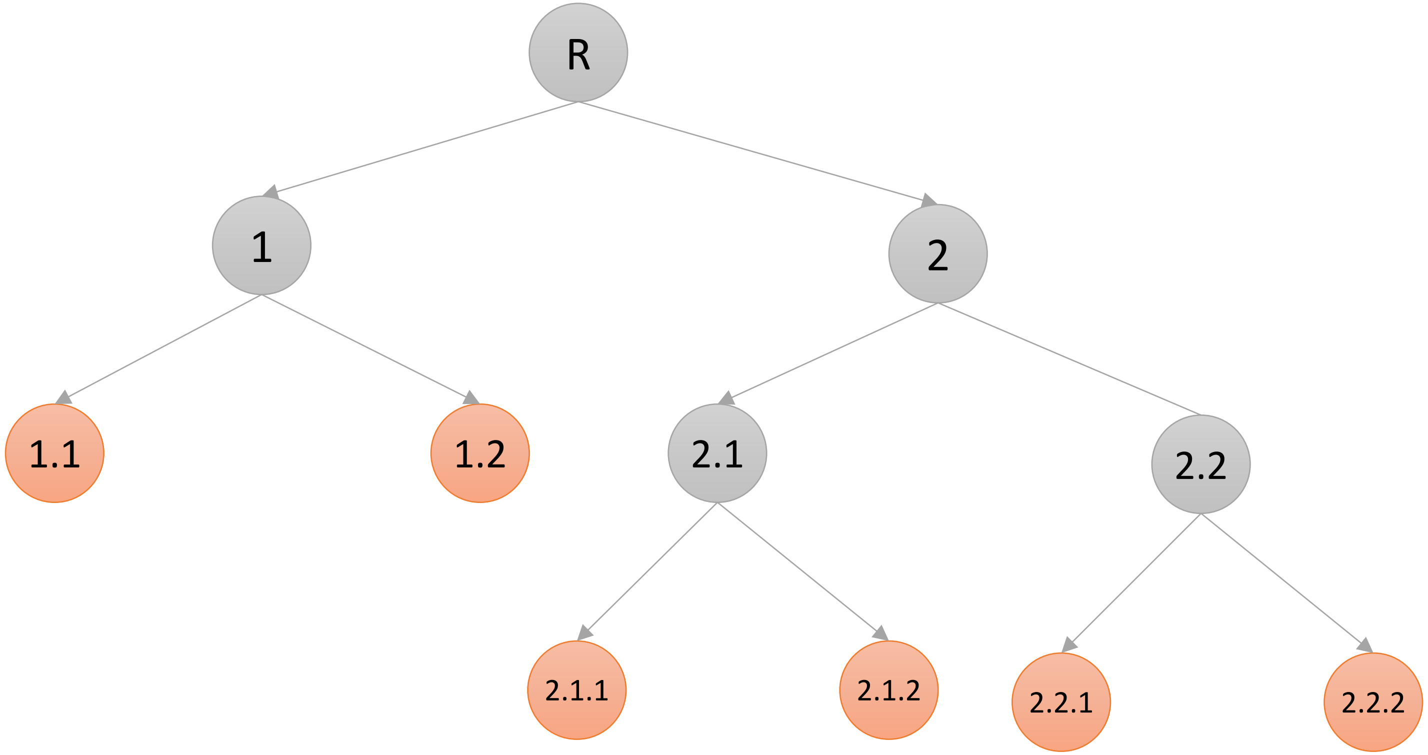
- Additionally it holds:
  1. Each node has exactly one parent and one label („Single-label NSHC“)
  2. Classification is done until we reach a leaf node

Different forms of NSHC

- In general we differ between three forms of NSHC:

    1. Flat Classification
    2. Local Methods
        1. Local classifier per node
        2. Local classifier per parent
        3. Local classification per layer
    3. Global Methods
        - Big-Bang approach

Different forms of NSHC: Flat classification



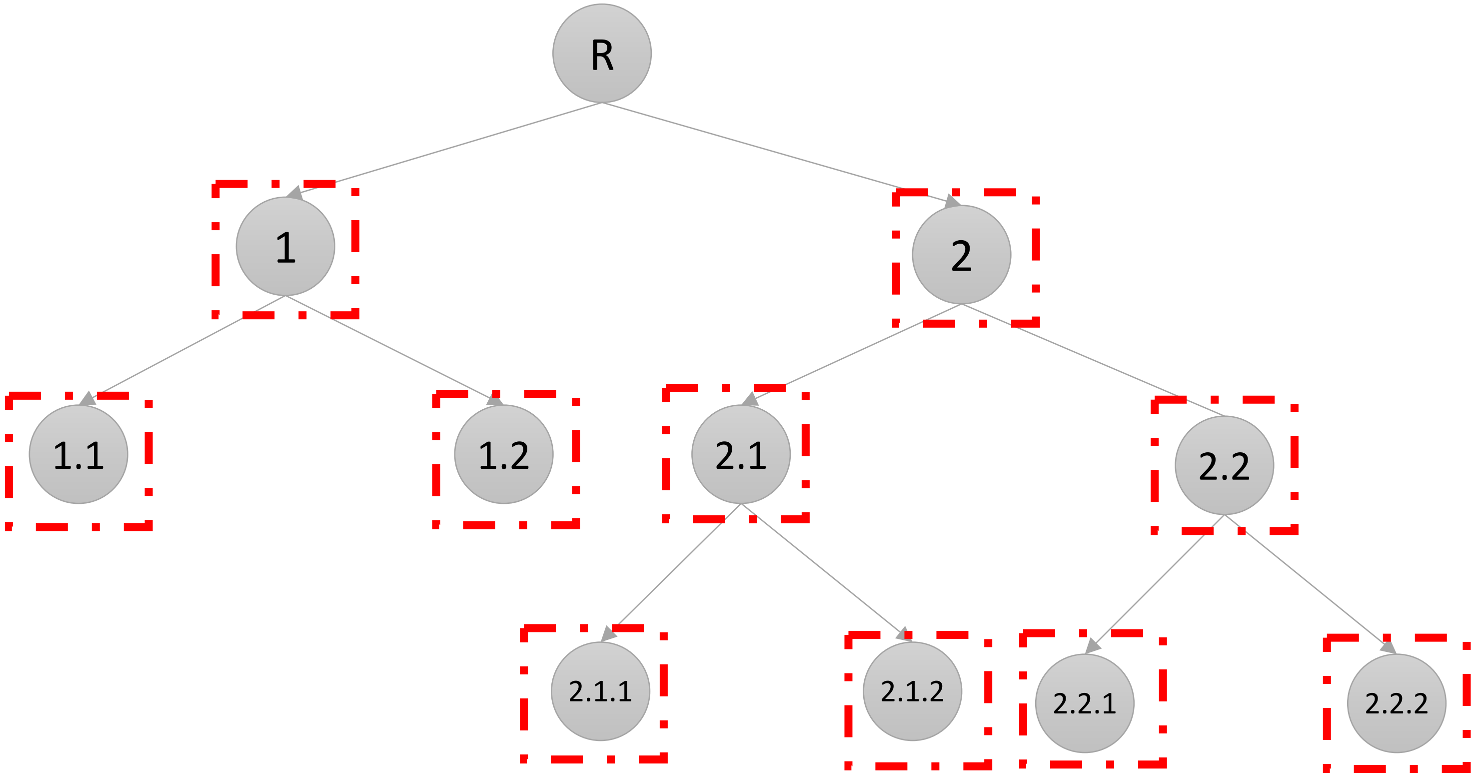
- Classification as you know it
- „Bottom-Up“, parent information can be inferred from the leavs
- Ignores the hierarchy information entirely

Forms of NSHC: Flat classification

- **Classifier:**
  - A single classifier
- **Training data:**
  - All instances, using the label according to the leavs

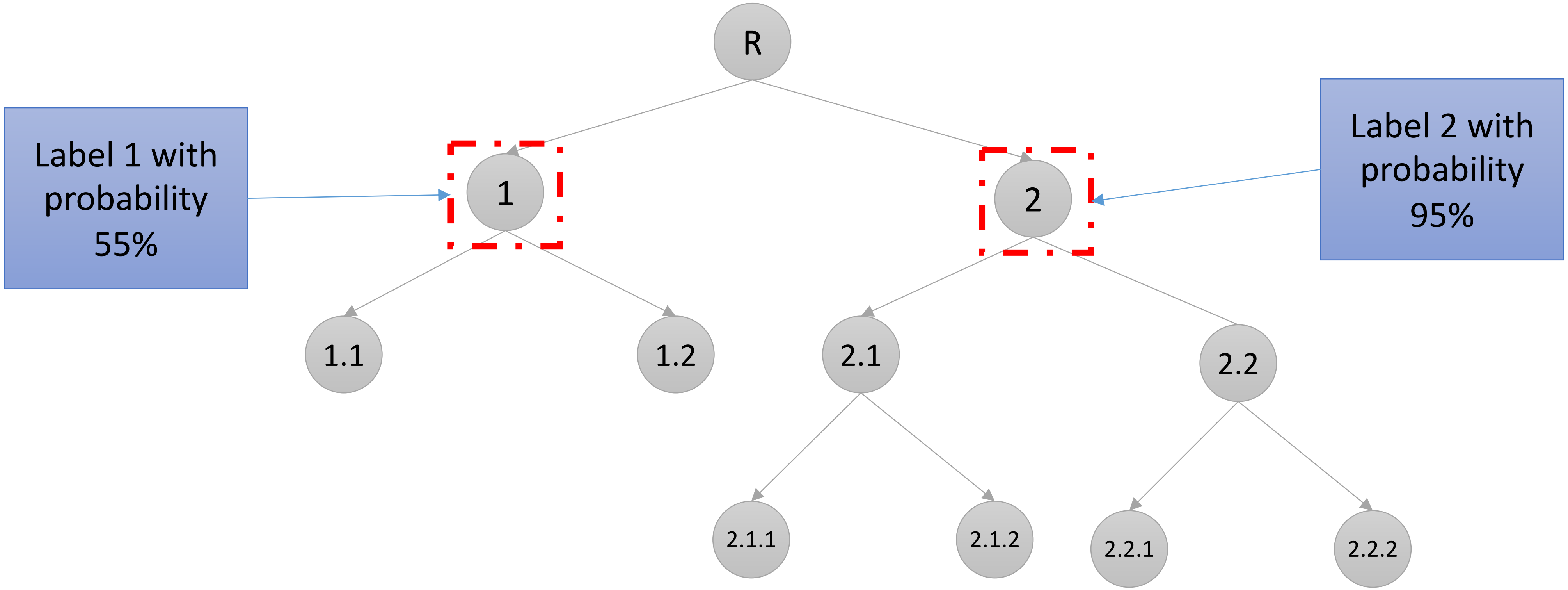
- **How to apply:**
  - Directly predict the most concrete label for every instance
  - Parents can be inferred from the leaf label if required
- **Problems:**
  - Does not make use of any structural information

Forms of NSHC: Local Classifier per Node (LCPN)

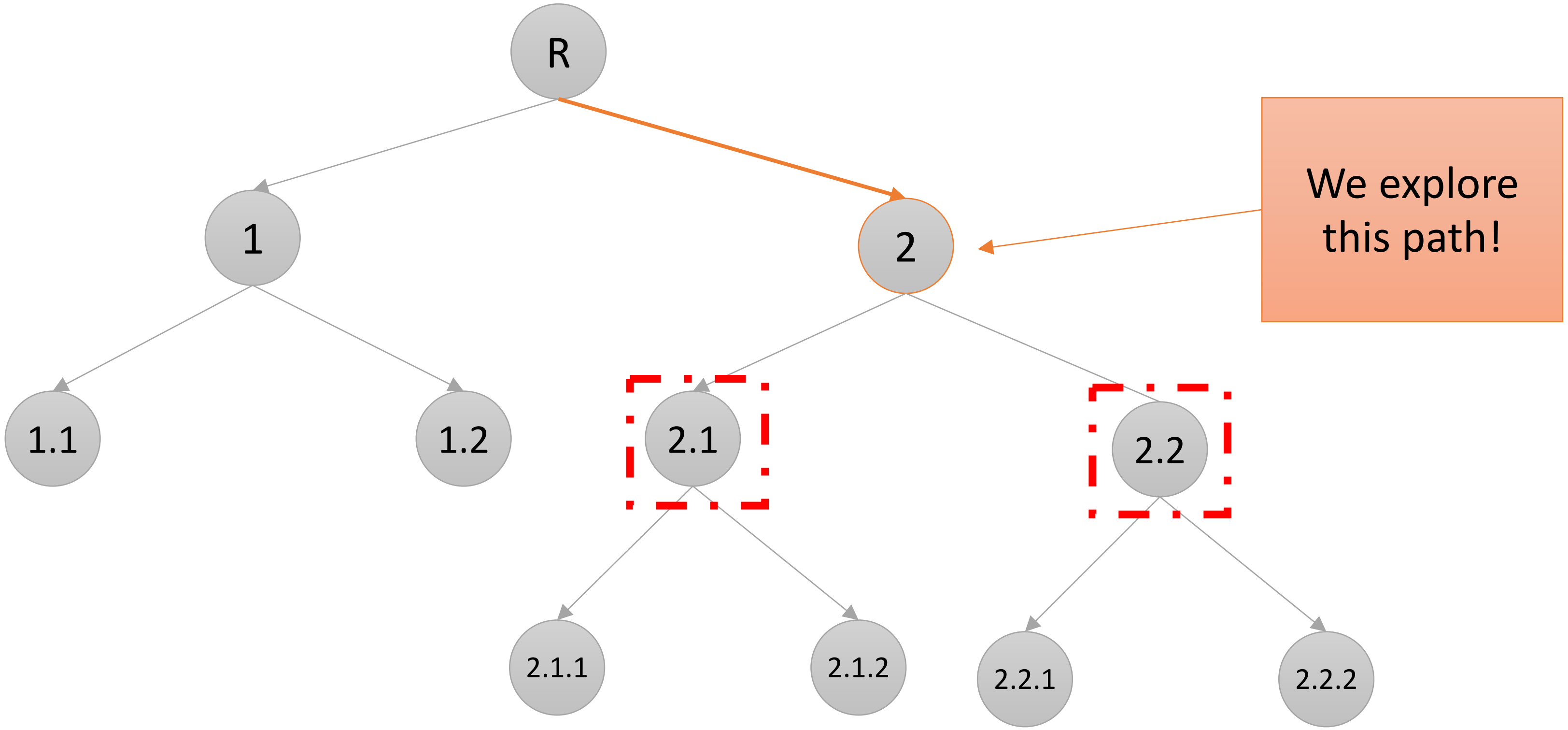


- Each node contains a classifier, which decides if the instance gets this label or not
- „Top-Down“ approach
- Most prominent approach in the literature (of NSHC)

Forms of NSHC: Local Classifier per Node (LCPN)



Forms of NSHC: Local Classifier per Node (LCPN)



Forms of NSHC: Local Classifier per Node (LCPN)

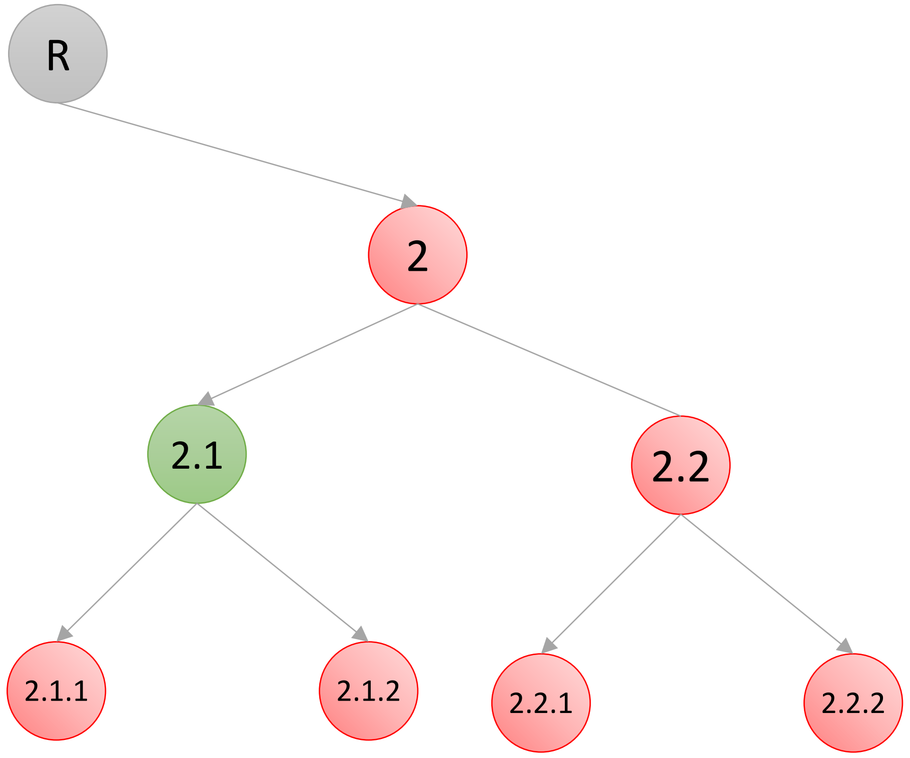
- **Classifier:**
  - One classifier for each label (or node)
- **Training:**
  - Multiple ways of training the individual classifiers (explained in coming slides)
    1. Exclusive-Policy
    2. Less-Exclusive-Policy
    3. Less-Inclusive-Policy
    4. Inclusive-Policy
    5. Siblings-Policy
    6. Exclusive-Siblings-Policy
  - Every classifier predicts a binary outcome („To be or not to be“)
- **How to apply:**
  - Evaluate all classifier separately and then use a strategy to get the final results!
    - We have shown a strategy where always the highest prediction wins
- **Problems:**
  - Requires a strategy to resolve inconsistencies
    - What if it is neither **1** or **2**

NSHC: LCPN training strategies

- For a local classifier, we can use a different subset of our training data:

1. Exclusive Policy

- **Positive Instances**
  - Only instance with finest class to be 2.1
- **Negative Instances**
  - All other instances

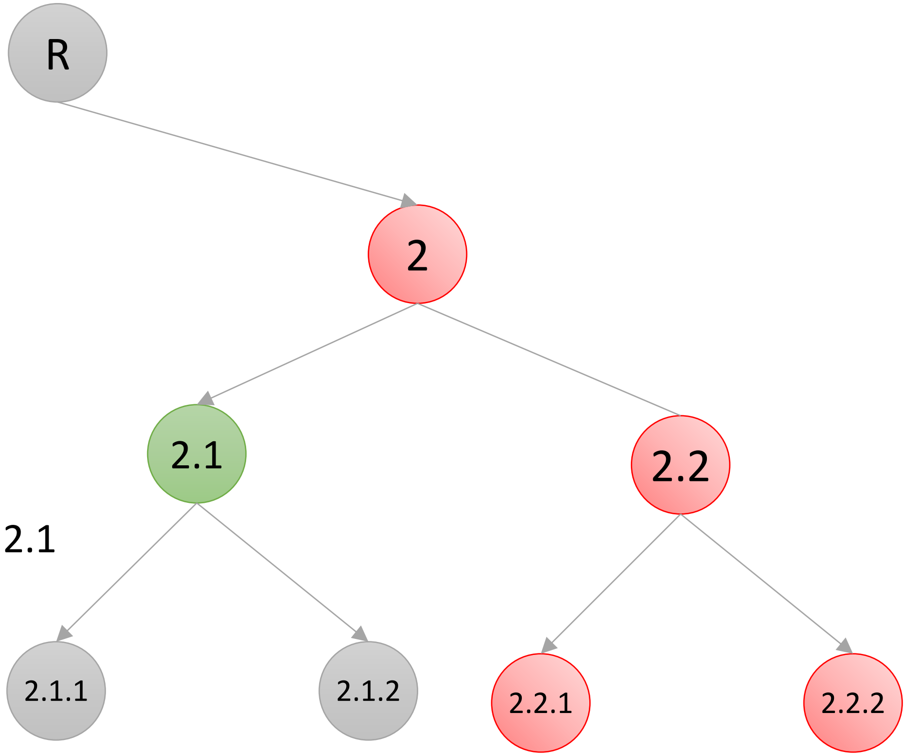


NSHC: LCPN training strategies

- For a local classifier, we can use a different subset of our training data:

2. Less-Exclusive Policy

- **Positive Instances**
  - Only instance with finest class to be 2.1
- **Negative Instances**
  - All instances that are no successor of 2.1 and are not labelled with 2.1

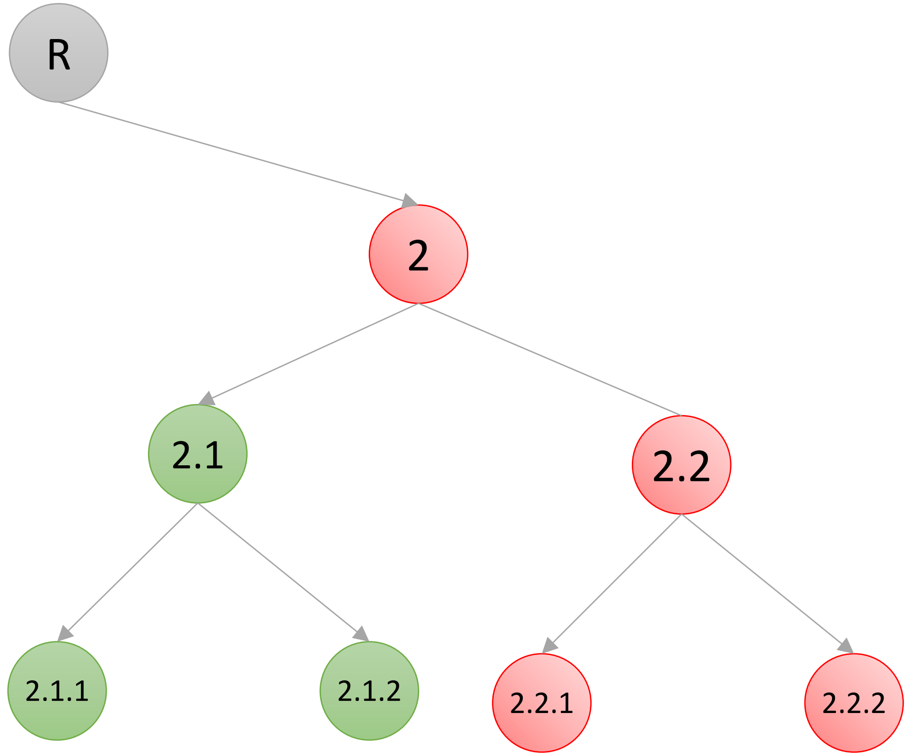


NSHC: LCPN training strategies

- For a local classifier, we can use a different subset of our training data:

3. Less-Inclusive Policy

- **Positive Instances**
  - All instances with label 2.1 or any successor (2.1.1 or 2.1.2)
- **Negative Instances**
  - All other instances

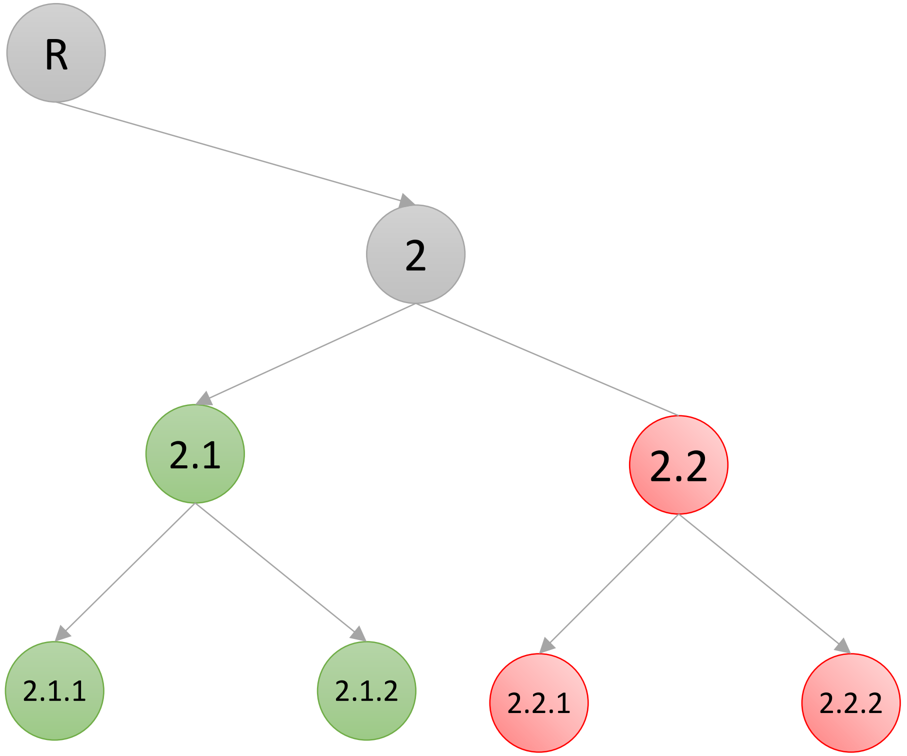


NSHC: LCPN training strategies

- For a local classifier, we can use a different subset of our training data:

4. Inclusive Policy

- **Positive Instances**
  - All instances with label 2.1 or any successor (2.1.1 or 2.1.2)
- **Negative Instances**
  - Alle instances except 2.1 as well as predecessors and successors (not 2.1, 2.1.1, 2.1.2 or 2).

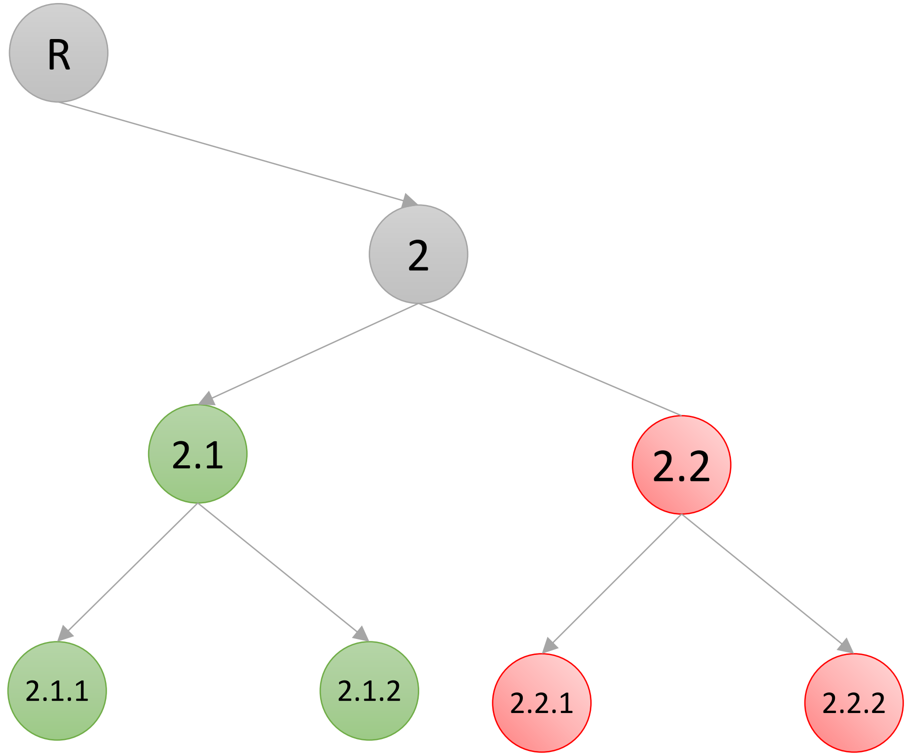


NSHC: LCPN training strategies

- For a local classifier, we can use a different subset of our training data:

5. Siblings Policy

- **Positive Instances**
  - All instances with label 2.1 or any successor (2.1.1 or 2.1.2)
- **Negative Instances**
  - All sibling-instances and their successors (2.2,2.2.1,2.2.2)

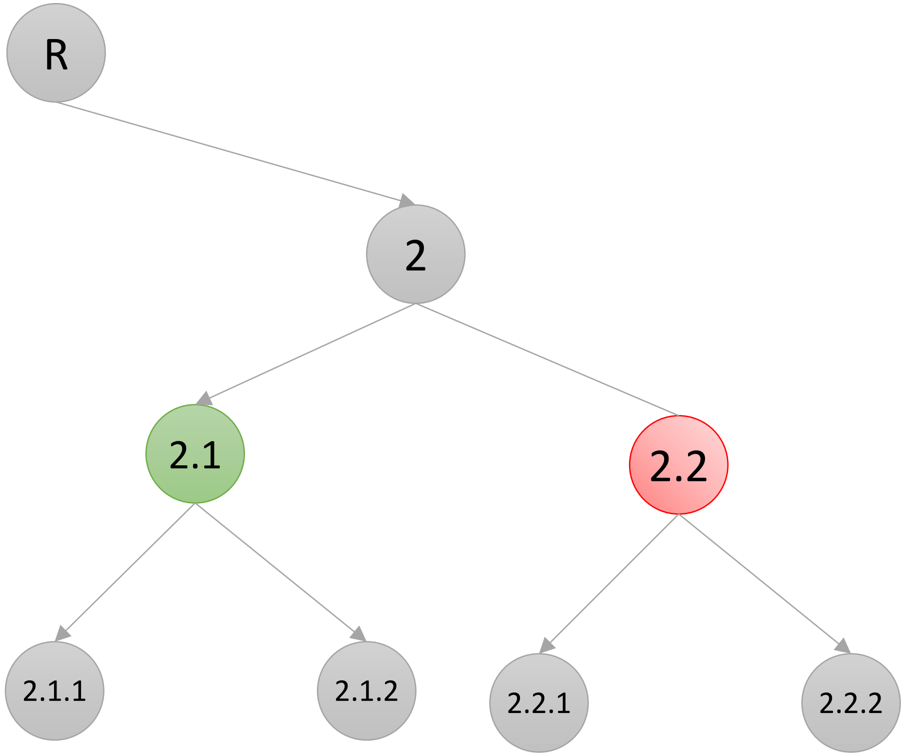


NSHC: LCPN training strategies

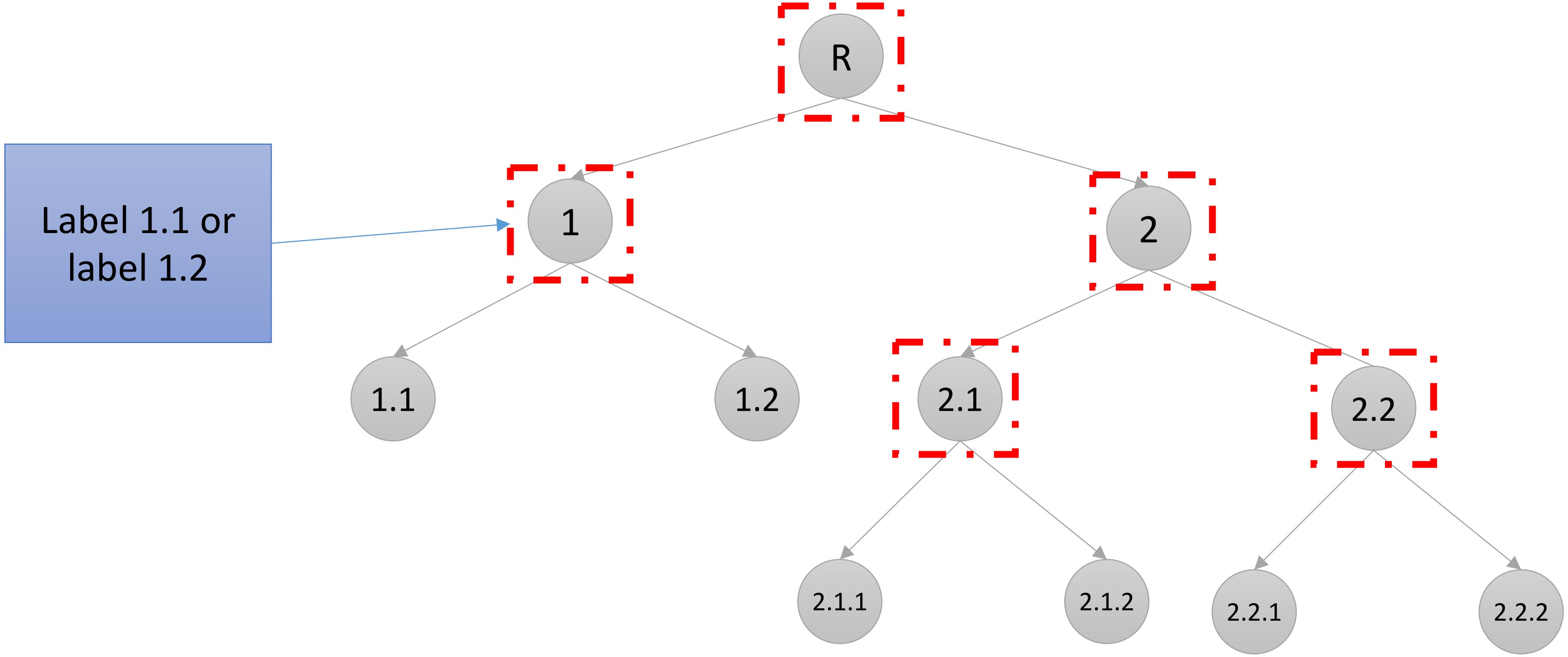
- For a local classifier, we can use a different subset of our training data:

6. Exclusive-Siblings Policy

- **Positive Instances**
  - Only instances with label 2.1
- **Negative Instances**
  - Only of sibling instances



Forms of NSHC: Local classifier per Parent Node (LCPN)

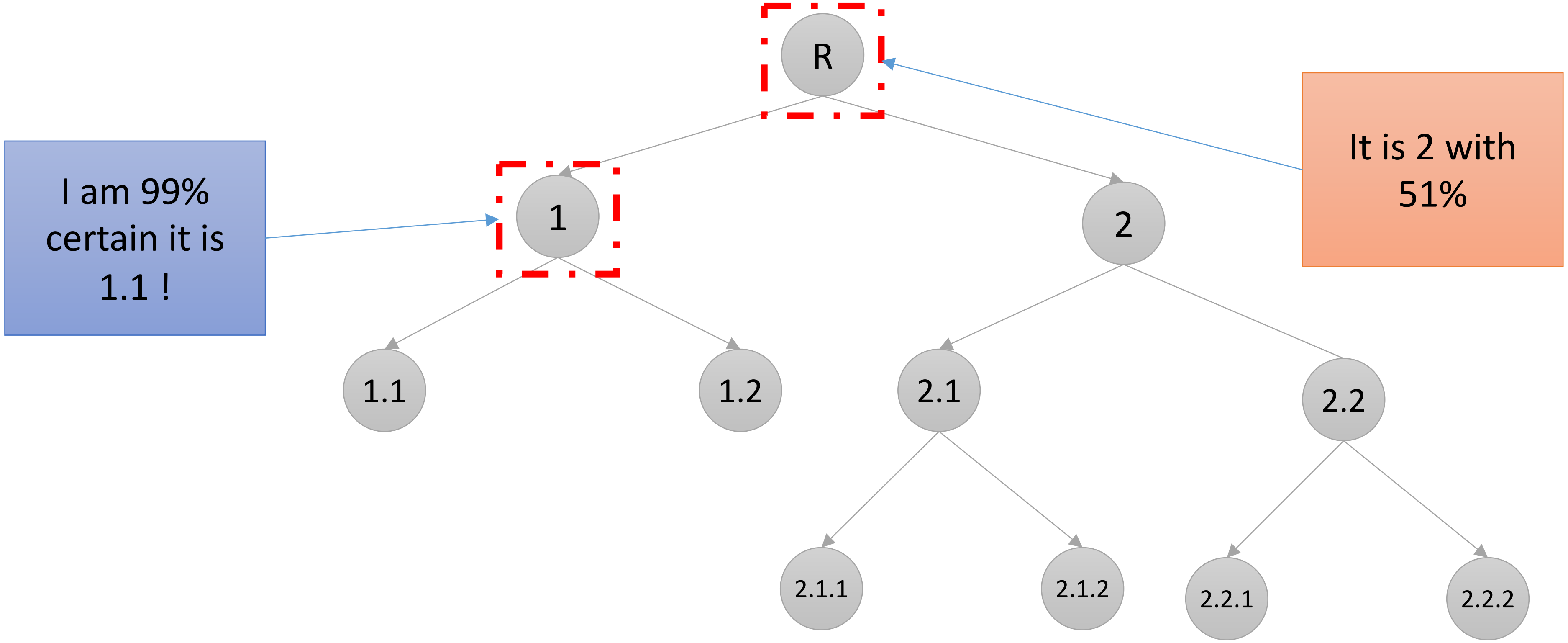


- One classifier per non-leaf node, that decides which child is more appropriate
- „Top-Down“ approach

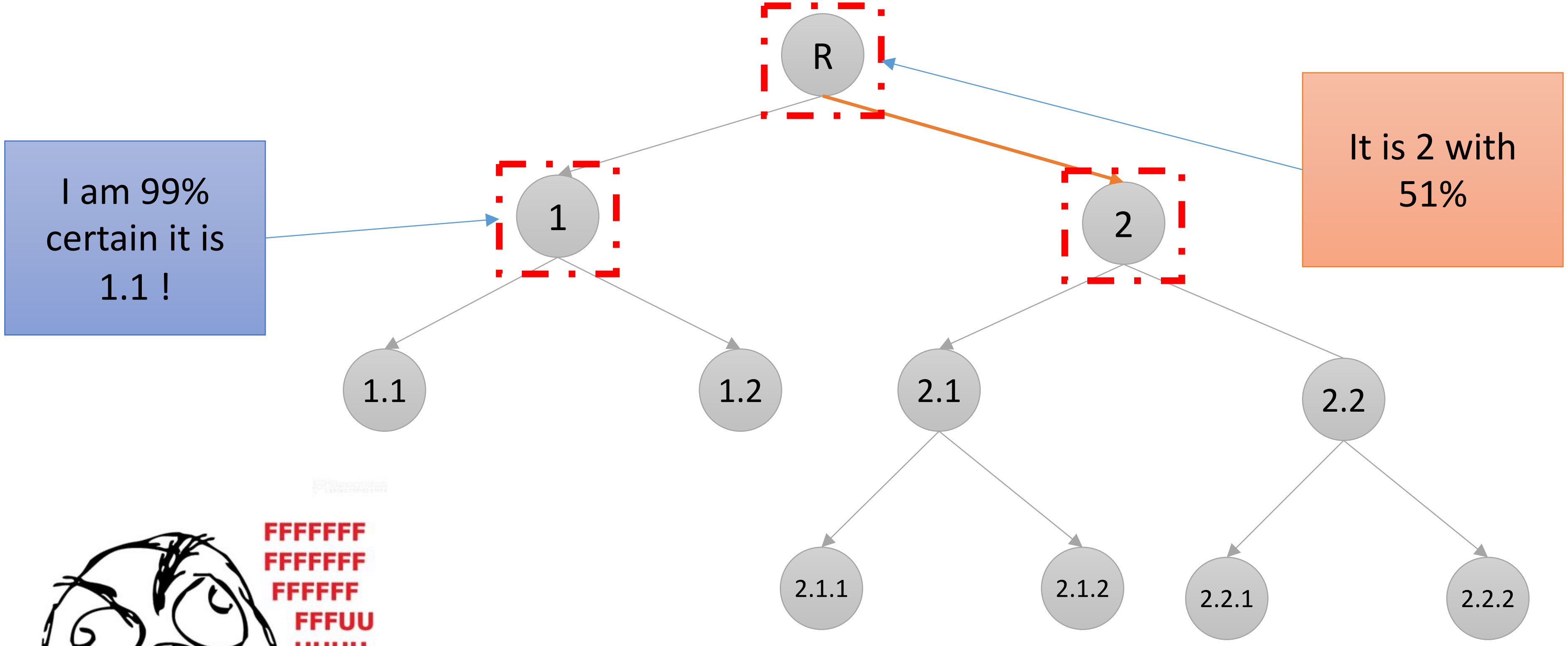
Forms of NSHC: Local classifier per Parent Node (LCPN)

- **Classifier:**
  - One classifier for every non-leaf node
- **Training:**
  - Multiple possibilities for the instances:
    1. Siblings-Policy
    2. Exclusive-Siblings-Policy
  - Multiclass prediction with respect to the children
- **How to apply:**
  - Query the classifier top-down and follow their most likely prediction
- **Probleme:**
  - (Potentially) requires an N-Ary classifier instead of just binary classifier
  - Might never reach the most appropriate classifier

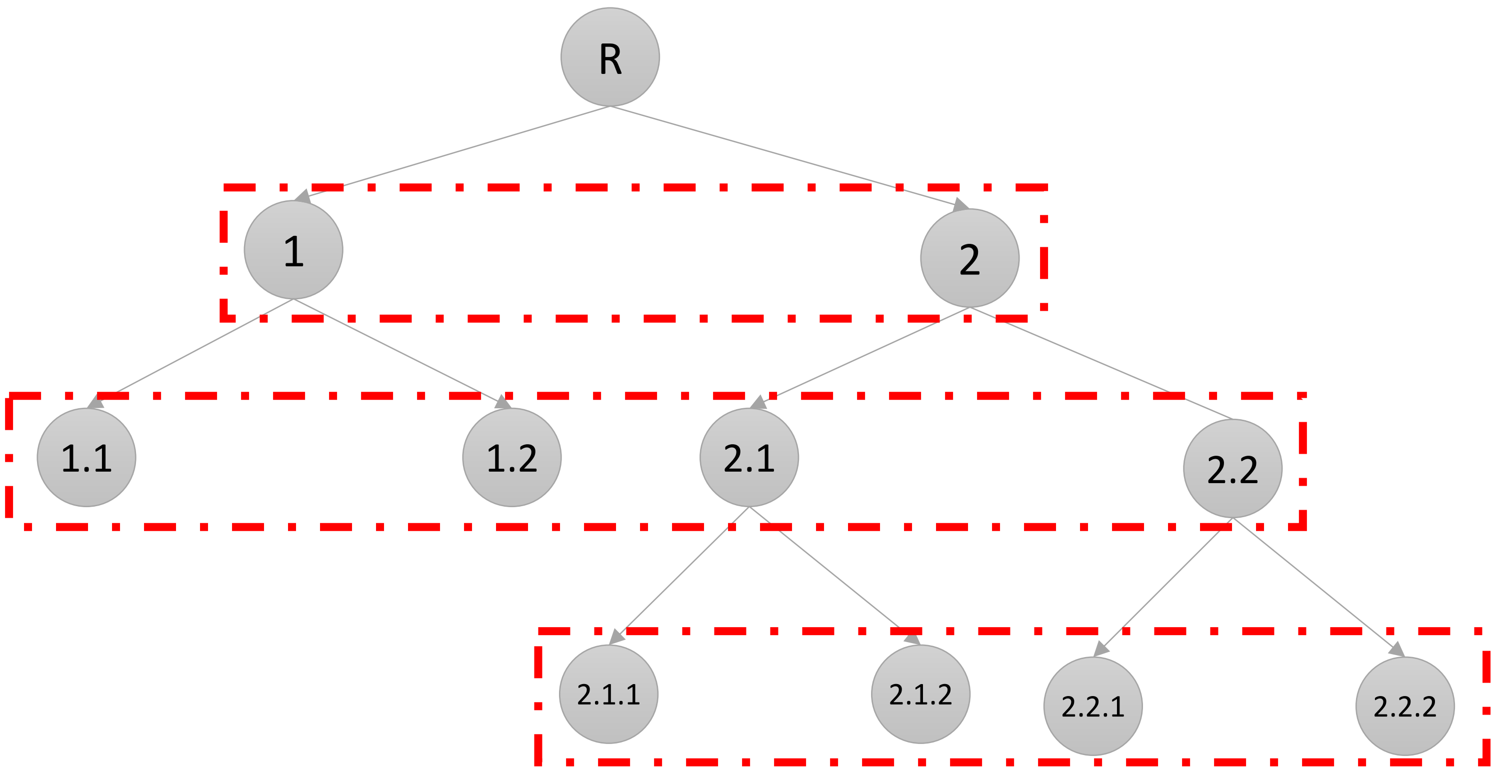
Forms of NSHC: Local classifier per Parent Node (LCPN)



Forms of NSHC: Local classifier per Parent Node (LCPN)



Formen von NSHK: Lokale Klassifikatoren pro Ebene

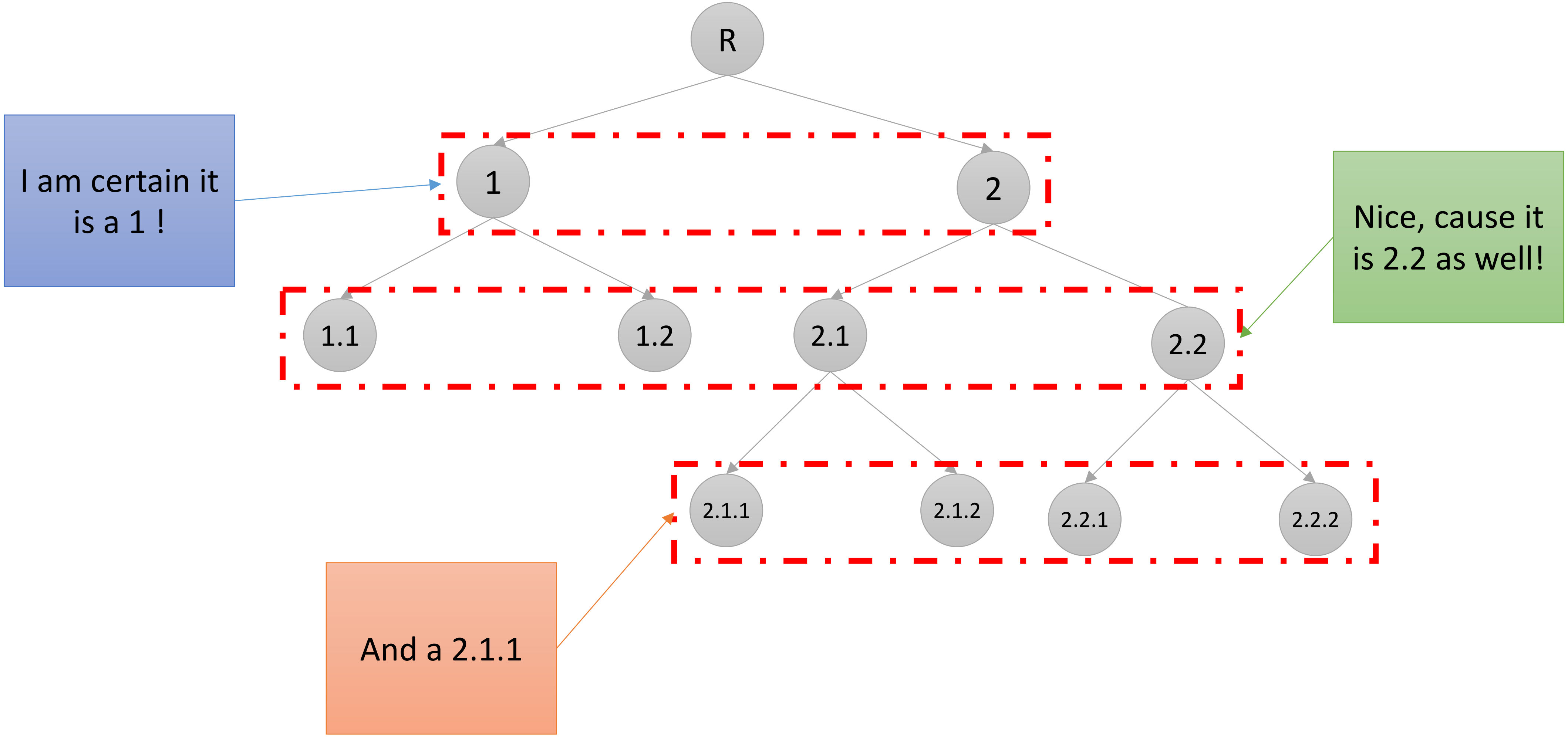


- Each layer has a distinct N-ary classifier
- Least used (in NSHC literature)
- Top-Down approach

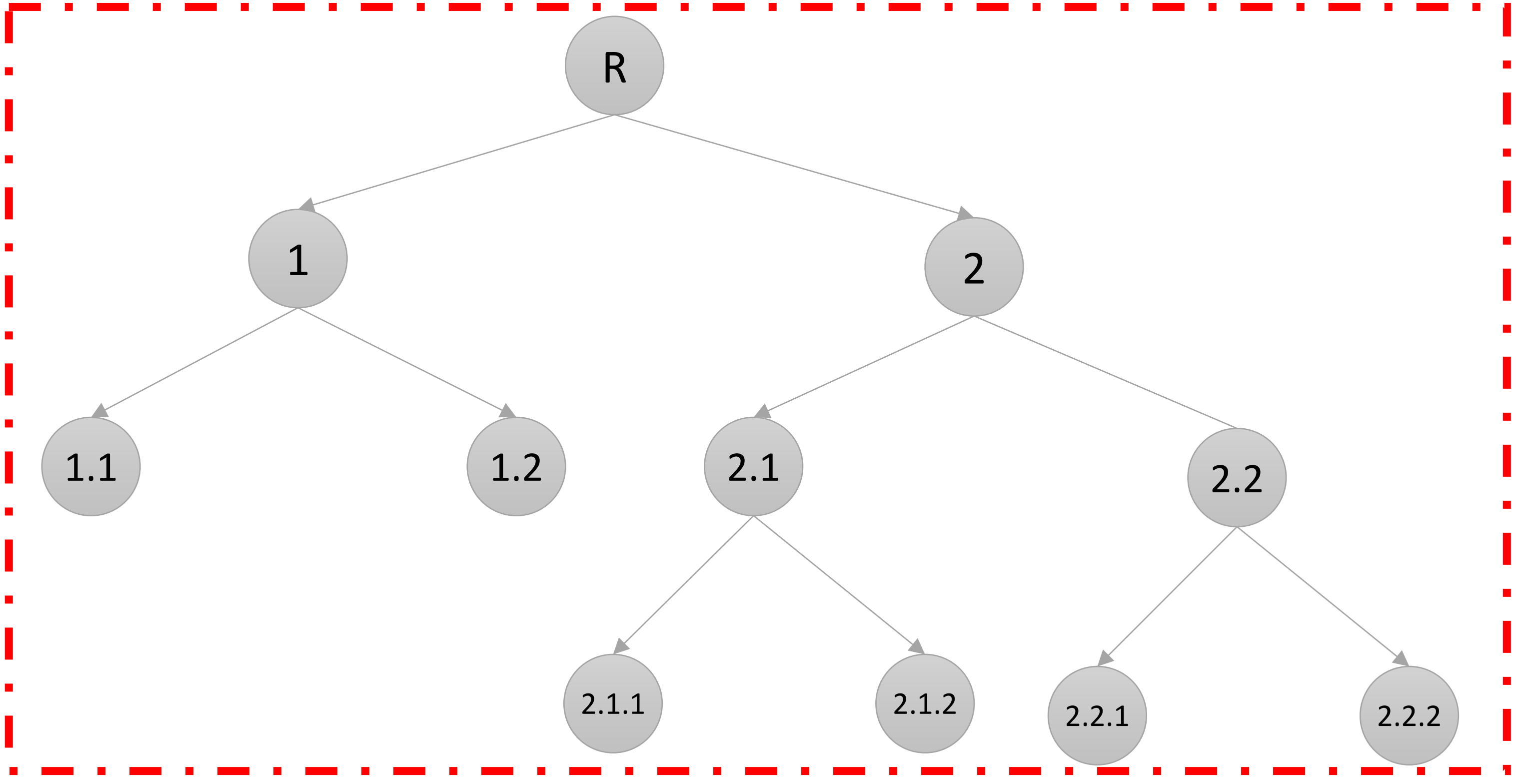
Forms of NSHC: Local classifier per Layer(LCL)

- **Classifier:**
  - One classifier per layer
- **Training:**
  - Again different possibilities for the training
    1. Siblings-Policy
    2. Exclusive-Siblings-Policy
  - Multiclass prediction at every layer.
- **How to apply:**
  - E.g. Apply top-down and follow their predictions
  - Or find the path with the highest score
- **Problems:**
  - Local inconsistencies

Forms of NSHC: Local classifier per Layer(LCL)



Forms of NSHC: Global Methods Big-Bang Approach



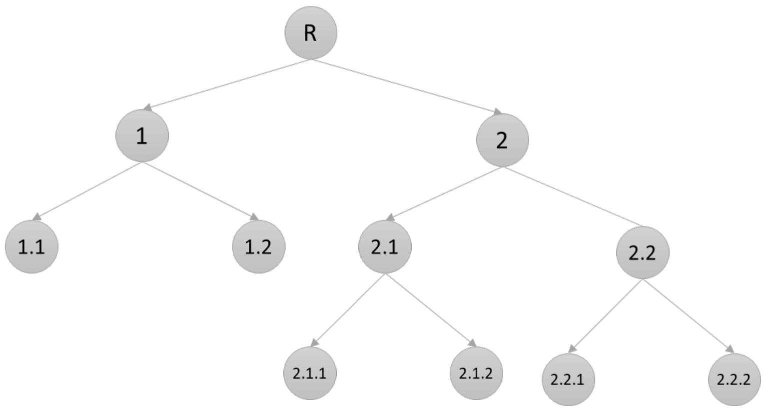
- A single classifier for the entire problem
- Makes fully use of the structure

Forms of NSHK: Global methods: Big-Bang approach

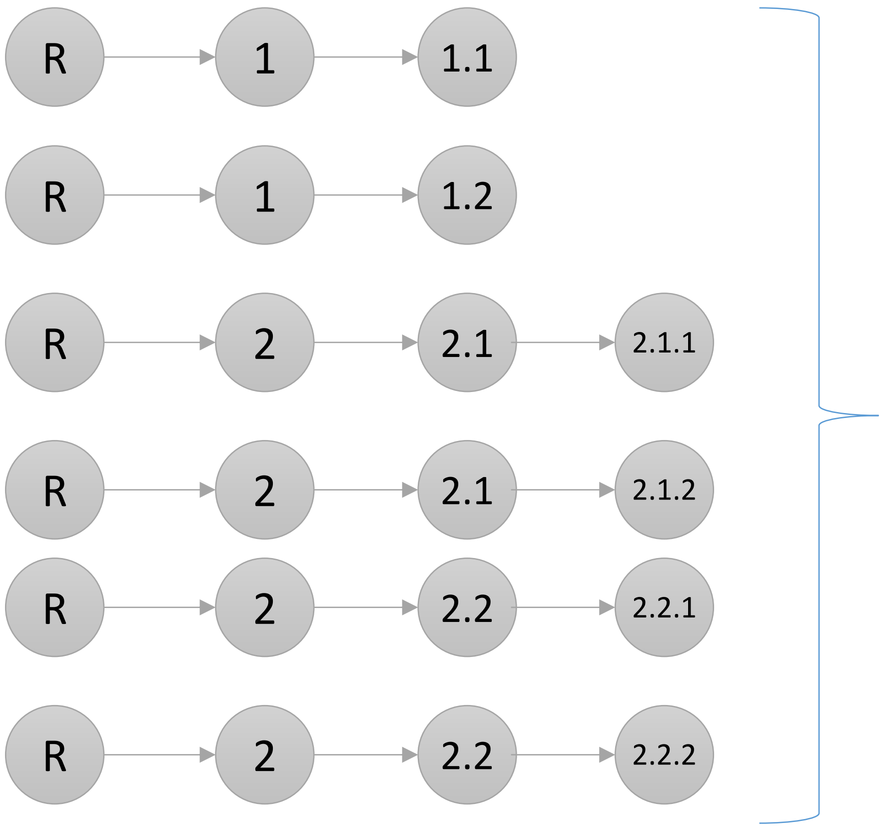
- **Classifier:**
  - A single classifier:
    - Structured Perceptron
    - Conditional Random Fields

- **Training:**
  - Instances have to be modelled according to the classifier
- **Application:**
  - Convert an instance appropriately and query the classifier
- **Problem:**
  - Needs dedicated algorithms!

Big Bang: Example

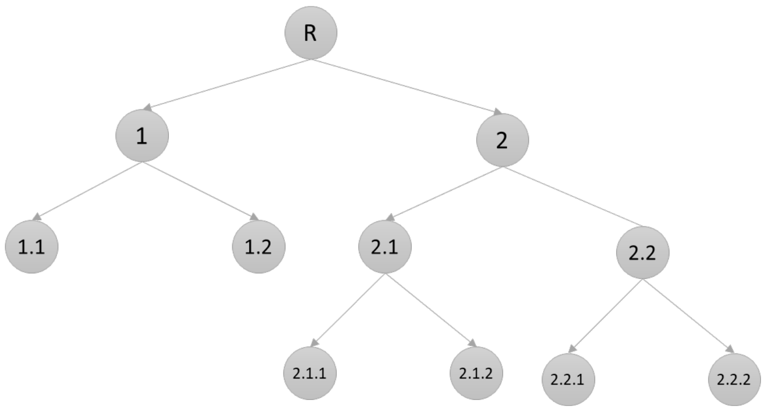


- Let us dissect our tree of labels into sequences



We basicly renamed our 6 leaf-labels into „path-labels“

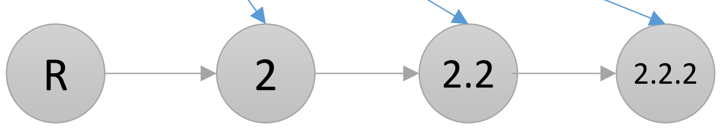
Big Bang: Example



- But we can then score more appropriately:

$$\underset{S \in valid\ Sequences}{argmax} \quad p(S|x) = \frac{\exp(\sum_{features:i} \lambda_i \cdot f(S,x))}{\sum_{S' \in valid\ Sequences} \exp(\sum_{features:i} \lambda_i \cdot f(S',x))}$$

- This means we can integrate features for every node on the path, similar to „back-off features“ that we introduced for CRFs



Further typical application scenarios

1. Music-Genre Prediction
2. Protein-function-prediction
3. Text classification
4. Object Detection
5. Predictions of Phonemes
6. … ➔ in any case a lot more than just relation classification!

Which NSHC is the best?

- No general answer, you probably have to try it!

Approach	Work	Result when compared against				
		Flat	LCN	LCPN	LCL	GC
LCN	Brecheisen et al. (2006a)	~				
	D'Alessio et al. (2000)	↑				
	Liu et al. (2005)	↑				
	Cesa-Bianchi et al. (2006a,b)	↑	↑			
	Cesa-Bianchi and Valentini (2009)	↑				
	DeCoro et al. (2007)	↑				
	Guan et al. (2008)	↑				

https://link.springer.com/content/pdf/10.1007/s10618-010-0175-9.pdf

➔ LCN seems to outperform flat classification

Which NSHK should I use?

Approach	Work	Result when compared against				
		Flat	LCN	LCPN	LCL	GC
LCPN	Koller and Sahami (1997)	~				
	Burred and Lerch (2003)	~				
	Chakrabarti et al. (1998)	↑				
	McCallum et al. (1998)	↑				
	Dumais and Chen (2000)	↑				
	Ruiz and Srinivasan (2002)	↑				
	Kriegel et al. (2004)	↑				

https://link.springer.com/content/pdf/10.1007/s10618-010-0175-9.pdf

➔ LCPN seems to outperform flat classification

Which NSHK should I use?

Approach	Work	Result when compared against				
		Flat	LCN	LCPN	LCL	GC
GC	Dekel et al. (2004a,b)	↑		↑		
	Wang et al. (2001)	↑				
	Peng and Choi (2005)	↑				
	Rousu et al. (2005, 2006)	↑				
	Blockeel et al. (2006)	↑				
	Cai and Hofmann (2004, 2007)	↑				
	Wang et al. (1999)	↑				
	Kiritchenko et al. (2005, 2006)	↑		~		

https://link.springer.com/content/pdf/10.1007/s10618-010-0175-9.pdf

➔ GC seems to outperform flat classification

Recap NSHC

- Applicable, if the labels are ordered in a hierarchy.
- Local Approaches
  - One classifier per node (LCN)
  - One classifier per parent node (LCPN)
  - One classifier per layer (LCL)

  ➔ Easy to apply, since any classifier can be used, but some approaches need strategies to resolve inconsistencies.
- Global Approaches
  - Big-Bang approach

  ➔ Classifier has to be able to deal with the structure

  ➔ Makes use of the entire hierarchy

➔ Results from literature show, that NSHC outperforms flat classification, but there is no clear winner!