Improving the Correctness of Medical Diagnostics Based on Machine Learning with Coloured Petri Nets

Irene Dovichi

Computational Models for Complex Systems

presentation a.y. 22/23

IN SUPREMÆ DIGNITATIS · 1343 ·
Università di Pisa

Introduction to the topic

- Breast cancer: most common cancer in women
- Importance of early detection of cancer
- Conventional methods of diagnosis
  - lab tests 🩸
  - imaging tests ☢️
  - biopsy 💉

  are exposed to human error, long-lasting, painful
- Alternative: Machine Learning based diagnosis 💻
- Formal verification of the prognostic process

Presentation scheme

1. Data collection
2. Formation of the decision tree
3. Transformation into an equivalent CP-Net
4. Incorporation in a Hierarchical CP-Net model
   - for correcting the rules
   - for proving the correctness of the model
5. Performance metrics

Dataset

<u>Breast Cancer Wisconsin Diagnostic (WDBC)</u>

- From the UCI Machine Learning Repository
- Digitized images of a FNA of a breast mass 💉
- 569 instances:
    - 212 malignant cases M ❌
    - 357 benign cases B ✔️
- The dataset was split in 2 groups:
    - training set 400 instances
    - testing set 169 instances
- 32 features: describe the cell nuclei present in the image 🧬

Dataset

Breast Cancer Wisconsin Diagnostic (WDBC): 32 features

- ID number
- Diagnosis (M/B)
- 10 real-valued features (Radius, Texture,...), for each of them:
  - mean value
  - standard error
  - worst value

Feature	Description	Mean	Standard error	Worst/largest value
Radius	Mean of the distances between the center and the points of the perimeter	6.98–28.11	0.11–2.87	7.93–36.04
Texture	Standard deviation of gray level values	9.71–39.28	0.36–4.89	12.02–49.54
Perimeter	The total distance between the snake points constitutes the nuclear perimeter	43.79–188.50	0.76–21.98	50.41–251.20
Area	Measured simply by counting the number of pixels on the interior of the snake	143.5–2501.00	6.80–542.20	185.20–4254.00
Smoothness	Local variation in radius lengths	0.05–0.16	0.00–0.03	0.07–0.22
Compactness	Perimeter^2/area - 1	0.02–0.35	0.00–0.14	0.03–1.06
Concavity	Severity of concave portions of the contour	0.00–0.43	0.00–0.40	0.00–1.25
Concave points	Number of concave portions of the contour	0.00–0.20	0.00–0.05	0.00-0.29
Symmetry	The major axis or longest chord through the center	0.11–0.30	0.01–0.08	0.16–0.66
Fractal dimension	Coastline approximation - 1	0.05–0.10	0.00–0.03	0.06–0.21

Machine learning algorithm

- We are interested in binary classification: M/B
- Methods:
  - decision trees
  - support vector machine
  - K-nearest neighbour
  - Naive Bayes
  - ...

Machine learning algorithm

- We are interested in binary classification: M/B
- Methods:
  - decision trees
  - support vector machine
  - K-nearest neighbour
  - Naive Bayes
  - ...

<u>C4.5 algorithm (J48) decision tree</u>

- Classification algorithm that produces decision trees based on information theory
- Best feature to split on: the one with greatest information gain

Information gain

$$\{A_1, \ldots, A_n\} = \text{set of features}, \ \{C_1, \ldots, C_k\} = \text{set of classes},$$
$$T = \text{set of training examples}$$

- Entropy of the training set

$$E(T) = -\sum_{i=1}^{k} \frac{|C_i|}{|T|} \cdot log_2 \frac{|C_i|}{|T|}$$

Information gain

$\{A_1, \ldots, A_n\}$ = set of features, $\{C_1, \ldots, C_k\}$ = set of classes,
T = set of training examples

- Entropy of the training set

$$E(T) = -\sum_{i=1}^{k} \frac{|C_i|}{|T|} \cdot log_2 \frac{|C_i|}{|T|}$$

- Conditional entropy of T given A_j

$$E(T|A_j) = \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot E(T_i)$$

where $\{T_1, \ldots, T_m\}$ are partitions of T derived from a certain condition on A_j

Information gain

- Entropy of the training set

$$E(T) = -\sum_{i=1}^{k} \frac{|C_i|}{|T|} \cdot log_2 \frac{|C_i|}{|T|}$$

- Conditional entropy of T given A_j

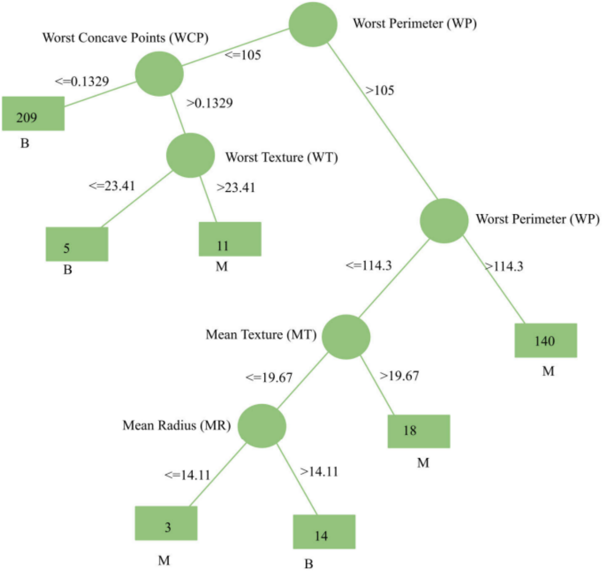
$$E(T|A_j) = \sum_{i=1}^{m} \frac{|T_i|}{|T|} \cdot E(T_i)$$

  where $\{T_1, \ldots, T_m\}$ are partitions of T derived from a certain condition on A_j

- Information gain of A_j

$$IG(T|A_j) = E(T) - E(T|A_j)$$

Decision tree trained on WDBC dataset



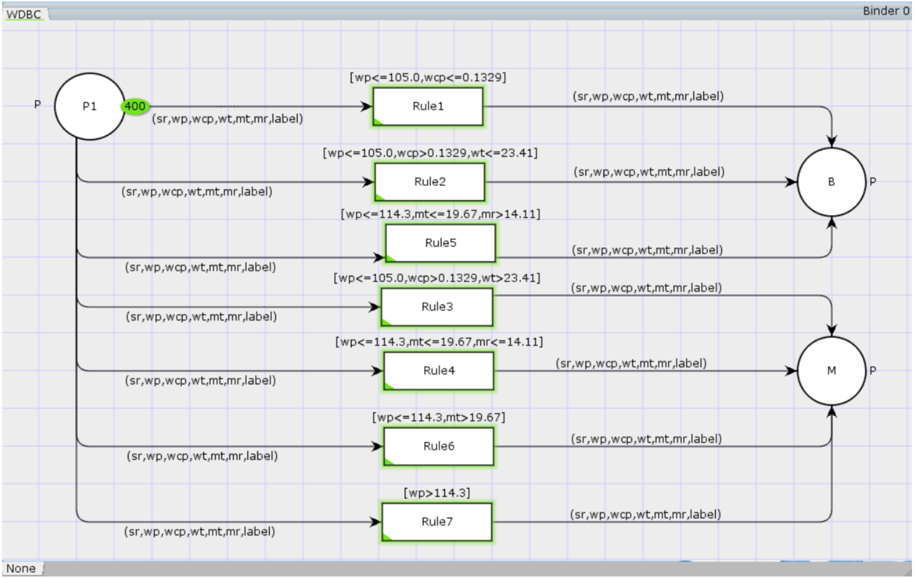
Prognostic rule extraction

7 leaf nodes 🍁 $\rightarrow$ 7 prognostic rules 📜

No	Label	Extracted rules	Rule nodes	Rule support	Rule accuracy
R1	B	Worst Perimeter(WP) $\leq$ 105 AND Worst Concave Points(WCP) $\leq$ 0.1329	2	209	98.08
R2	B	Worst Perimeter(WP) $\leq$ 105 AND Worst Concave Points(WCP) >0.1329 AND Worst Texture(WT) $\leq$ 23.41	3	5	100
R3	M	Worst Perimeter(WP) $\leq$ 105 AND Worst Concave Points(WCP) >0.1329 AND Worst Texture(WT) >23.41	3	11	90.90
R4	M	Worst Perimeter(WP) $\leq$ 114.3 AND Mean Texture(MT) $\leq$ 19.67 AND Mean Radius(MR) $\leq$ 14.11	3	173	5.20
R5	B	Worst Perimeter(WP) $\leq$ 114.3 AND Mean Texture(MT) $\leq$ 19.67 AND Mean Radius(MR) >14.11	3	19	100
R6	M	Worst Perimeter(WP) $\leq$ 114.3 AND Mean Texture(MT) >19.67	2	68	38.23
R7	M	Worst Perimeter(WP) >114.3	1	140	98.57

- Rule support: number of instances that comply with rule conditions
  - $\hookrightarrow$ threshold value: 5
- Rule accuracy: ratio of correctly classified instances

CP-Net transformation



Coloured Petri Nets - Intuition

- Adding colours to the <u>tokens</u> 🎨🖌️
  - to distinguish between different types of tokens
  - to attach a data value to them
- Each <u>place</u> is associated with a set of colours 🏷️
- <u>Transitions</u> are aware of the colours 👀
  - tokens may change colour when a transition fires
    $\hookrightarrow$ arc expressions
  - tokens may have to satisfy a certain condition
    $\hookrightarrow$ guards
- Generalize Petri nets 🕸️
  - from CP-Nets to PT-Nets uniquely
  - from PT-Nets to CP-Nets in many ways

Coloured Petri Nets - Formal definition

A Coloured Petri Net (CP-Net) is a tuple

$$N = (Pl, Tr, F, \Sigma, C, g, f, M_0)$$

where:

- Pl non-empty finite set of places
- Tr non-empty finite set of transitions
- F finite set of arcs
- Σ finite set of non-empty colour sets
- $C : Pl \to \Sigma$ colour function
- $g : Tr \to EXP$ guard function
- $f : F \to EXP$ arc expression function
- $M_0 : Pl \to EXP$ initialization function

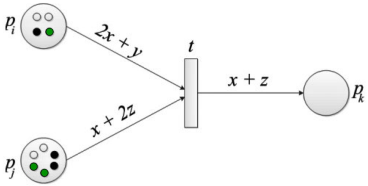
and: $Pl \cap Tr = Pl \cap F = Tr \cap F = \varnothing$

Coloured Petri Nets - Behaviour

Dynamic behaviour of CP-Nets is observed by firing transitions 🔥

- A transition can be fired only if it is enabled
- The firing of a transition creates a new marking

Example



- Initial marking

$$m(p_i) = 2'W + 1'B + 1'G$$
$$m(p_j) = 2'W + 2'B + 2'G$$

- Possible bindings for m

$$b_1 = \langle x = W,\ y = B,\ z = G \rangle$$
$$b_2 = \langle x = W,\ y = G,\ z = B \rangle$$

- New marking

$$m_1(p_k) = 1'W + 1'G$$
$$m_2(p_k) = 1'W + 1'B$$

Hierarchical CP-Nets

A Hierarchical Coloured Petri Net (HCP-Net) is a finite set of nets

$$HCPN = \{S_1, S_2, \ldots\}$$

and each S_i is a tuple

$$S_i = (Pl, Tr, F, \Sigma, C, g, f, M_0, h)$$

where:

- F, Σ, C, g, f, M_0 defined as in CP-Nets
- $Pl = ODP \cup INP$ set of ordinary places (as in CP-Nets) and interface places
- $Tr = ODT \cup HPT$ set of ordinary transitions (as in CP-Nets) and hyper-transitions
- h function that maps each hyper-transition to a net

CPN Tools

- CPN Tools is a computer tool for editing, simulating, and analyzing high-level Petri nets
- Two main components
  - graphical editor 🖱️
  - backend simulator (CPN ML) ⌨️
- It supports basic, timed, and coloured Petri nets
- Advantages
  - better results for both graphics and calculations
  - faster results
  - interactive presentations

Hierarchical CP-Net for medical diagnostics

Color Set Definition	Description
colset NO = INT;	Specifies the serial number for patient record.
colset WP = REAL;	Specifies the Worst Perimeter.
colset WCP = REAL;	Specifies the Worst Concave Points.
colset WT = REAL;	Specifies the Worst Texture.
colset MT = REAL;	Specifies the Mean Texture.
colset MR = REAL;	Specifies the Mean Radius.
colset L = string;	Specifies the patient record classification label.
colset P = product NO*WP*WCP*WT*MT*MR;	Cartesian product specifies the patient record attributes present in feature list.
colset P_LAB = product NO*WP*WCP*WT*MT*MR*L;	Cartesian product specifies the patient record with class label attribute.
colset P_ACL_PRL = product NO*WP*WCP*WT*MT*MR*L*L;	Cartesian product specifies the patient record with actual and predicted class label attributes.
colset ACTUAL_LAB = product NO*L;	Cartesian product specifies the actual label for a patient record.

Figure: Declaration of colour sets

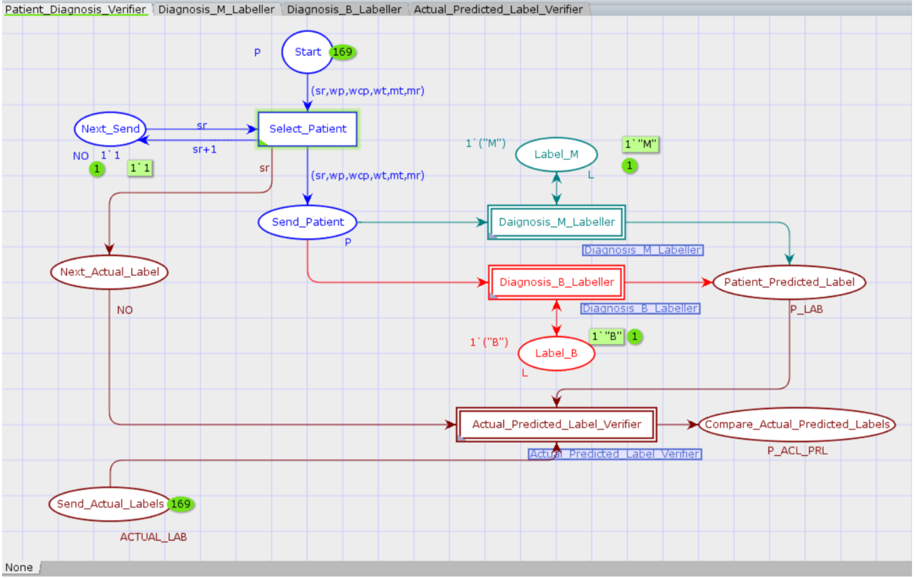
Hierarchical CP-Net for medical diagnostics

Algorithm 1 Decision Rules to CP-Nets

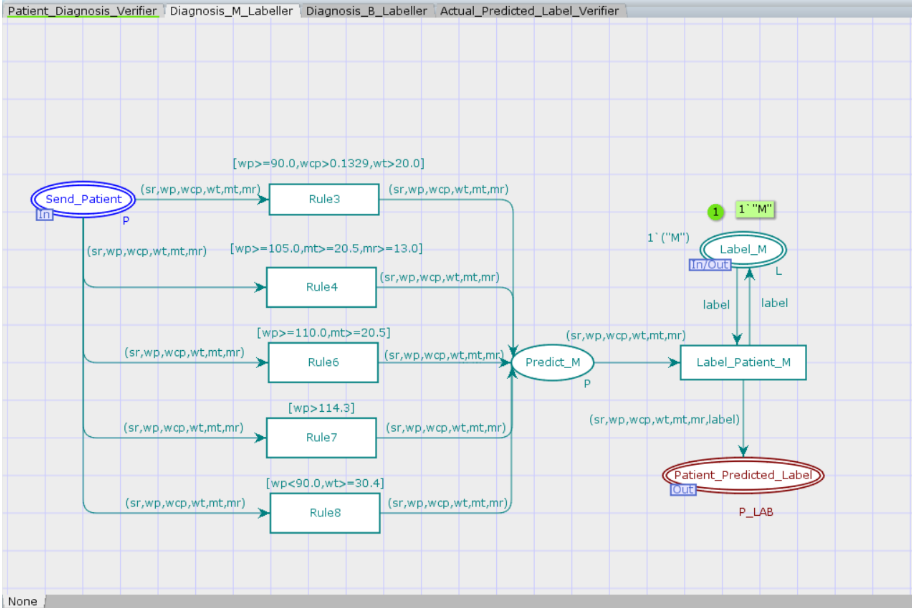
```
Input: Rule Set and Class Label
Output: CP-Net Submodule
for decision rule in Rule Set do
    Create CP-Net Transition
    for rule.attributes in rule do
        Add transition guard for rule.attribute
    end
end
```

Figure: Pseudocode for creating submodules

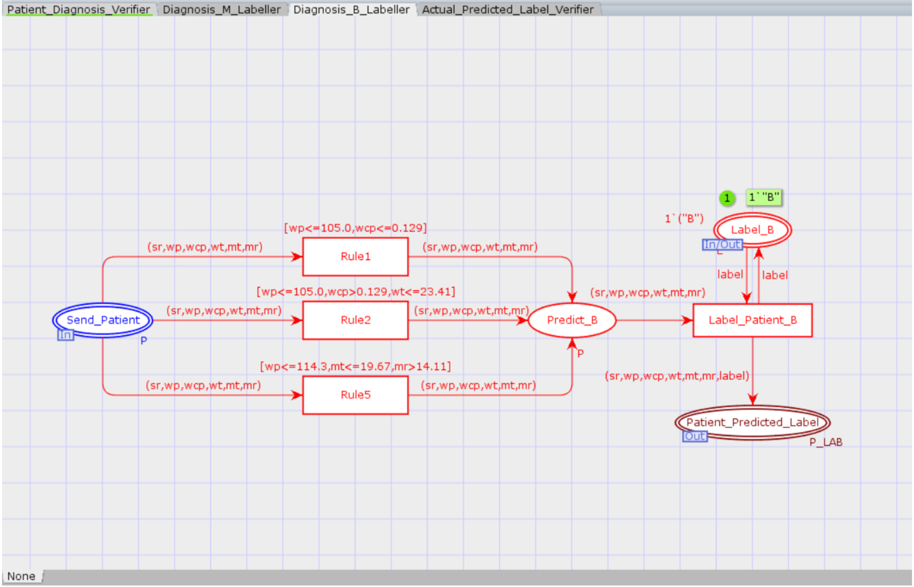
Hierarchical CP-Net for medical diagnostics



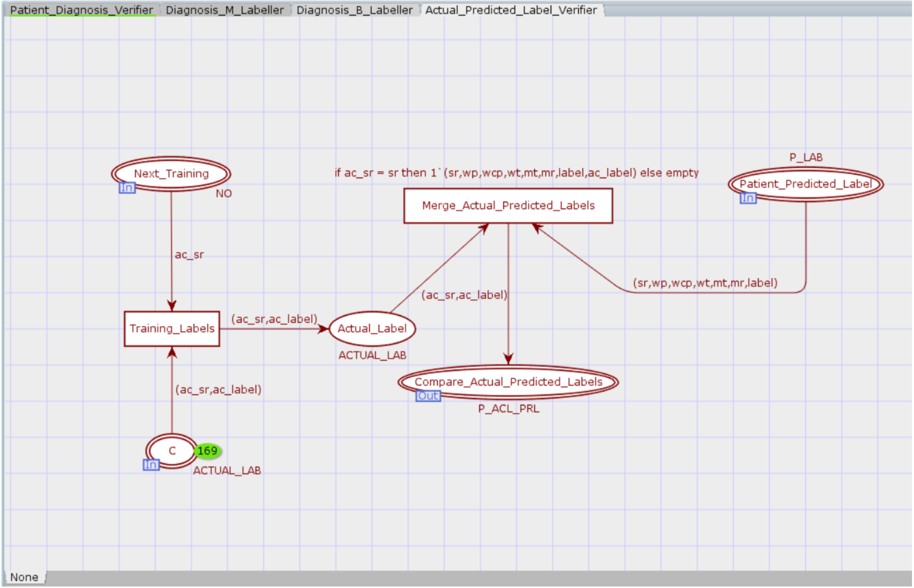
Diagnosis M submodule



Diagnosis B submodule



Verifier submodule



State spaces

- Useful tool to study the structural and dynamic properties of a CP-Net
- Calculate all reachable states (markings) and state changes (occurring binding elements) of the CP-Net
- Represent them in a directed graph
  - the nodes are the reachable markings
  - the arcs correspond to occurring binding elements
- State space explosion ⚠️

Correctness algorithm

- The positions of the tokens representing patients are analysed in the state space generated by the CPN Tools
- Checking the place *Compare_Actual_Predicted_Labels* we can find the patients that have been misclassified
- In case of misclassification rules are revised or possibly a new rule is created so that this instance is correctly classified
- The process continues until all rules are adjusted to pass the maximum number of tokens

Correctness algorithm

Algorithm 2: Decision Rule Correctness Process

```
Input: CP-Nets Model and Token Set
Output: Formally Verified CP-Nets Model
for Token in TokenSet do
    for Rules in CP-NetsModel do
        if State.M ← Token then
            if Token.label == M then
                Rules ← Verified
            else
                Rules ← ReviseRule(Rules,Token)
            end
        else
            if Token.label == B then
                Rules ← Verified
            else
                Rules ← ReviseRule(Rules,Token)
            end
        end
    end
end
```

Figure: Pseudocode to correct the model

```
Function ReviseRule(RuleSet, Token)
for Rule in RuleSet do
    for Attributes in Rule do
        Mathematical adjustments
    end
    if Token is simulated successfully then
        return true
    else
        Add new Rule into RuleSet
    end
end
```

Figure: ReviseRule subroutine

Revised rules

🎉 Decision rules after implementing Algorithm 2 🎉

No	Label	Extracted rules	Remarks	Rule support	Rule accuracy
R1	B	Worst Perimeter(WP) $\leq$ 105 AND Worst Concave Points(WCP) $\leq$ 0.129	Rule Revised	105	100
R2	B	Worst Perimeter(WP) $\leq$ 105 AND Worst Concave Points(WCP) $>$ 0.129 AND Worst Texture(WT)$\leq$ 23.41	Rule Revised	/	/
R3	M	Worst Perimeter(WP) $\geq$ 90 AND Worst Concave Points(WCP) $>$ 0.1329 AND Worst Texture(WT)$>$ 20	Rule Revised	43	83.72
R4	M	Worst Perimeter(WP) $\geq$ 105 AND Mean Texture(MT) $\geq$ 20.5 AND Mean Radius(MR)$\geq$ 13	Rule Revised	33	84.84
R5	B	Worst Perimeter(WP)$\leq$ 114.3 AND Mean Texture(MT)$\leq$ 19.67 AND Mean Radius(MR)$>$ 14.11	No Changes	13	92.30
R6	M	Worst Perimeter(WP)$\geq$ 110 AND Mean Texture(MT)$\geq$ 20.5	Rule Revised	30	90
R7	M	Worst Perimeter(WP)$>$ 114.3	No Changes	36	91.66
R8	M	Worst Perimeter(WP)$<$ 90 AND Worst Texture(WT)$\geq$ 30.4	New Rule	/	/

Revised rules

🎉 Decision rules after implementing Algorithm 2 🎉

No	Label	Extracted rules	Remarks	Rule support	Rule accuracy
R1	B	Worst Perimeter(WP) $\leq$ 105 AND Worst Concave Points(WCP) $\leq$ 0.129	Rule Revised	105	100
R2	B	Worst Perimeter(WP) $\leq$ 105 AND Worst Concave Points(WCP) $>$ 0.129 AND Worst Texture(WT)$\leq$ 23.41	Rule Revised	/	/
R3	M	Worst Perimeter(WP) $\geq$ 90 AND Worst Concave Points(WCP) $>$ 0.1329 AND Worst Texture(WT)$>$ 20	Rule Revised	43	83.72
R4	M	Worst Perimeter(WP) $\geq$ 105 AND Mean Texture(MT) $\geq$ 20.5 AND Mean Radius(MR)$\geq$ 13	Rule Revised	33	84.84
R5	B	Worst Perimeter(WP)$\leq$ 114.3 AND Mean Texture(MT)$\leq$ 19.67 AND Mean Radius(MR)$>$ 14.11	No Changes	13	92.30
R6	M	Worst Perimeter(WP)$\geq$ 110 AND Mean Texture(MT)$\geq$ 20.5	Rule Revised	30	90
R7	M	Worst Perimeter(WP)$>$ 114.3	No Changes	36	91.66
R8	M	Worst Perimeter(WP) $<$ 90 AND Worst Texture(WT)$\geq$ 30.4	New Rule	/	/

- They correspond to the guards that appeared in the M and B submodules
- Rules 2 and 8 are discarded since they have support < 5

Accuracy comparison

No	Before	After
R1	98.08	100
R3	90.90	83.72
R4	5.20	84.84
R5	100	92.30
R6	38.23	90
R7	98.57	91.66
Average	**71.83**	**90.42**

Figure: Rule accuracy before and after Algorithm 2

State space report

CPN Tools allows you to save state space analysis calculations

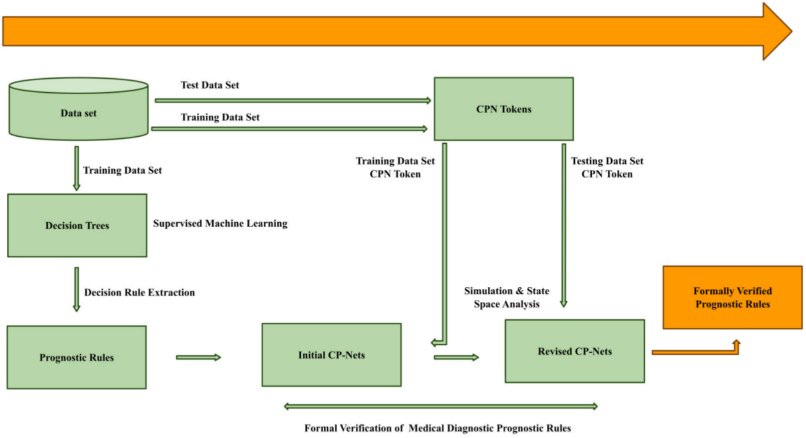
- Strongly-connected-component graph (SCC graph) 🔗
  - the nodes are subgraphs of the state space
  - state space nodes n_1 and n_2 are in the same subgraph iff they are mutually reachable
- Home marking can be reached from any reachable marking 🏠
- Dead marking when no transition is enabled ☠️

State space report of our model

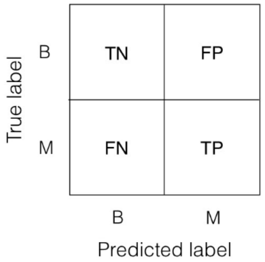
```
 Statistics
---------------------

  State Space
     Nodes:  28501
     Arcs:   138718
     Secs:   300
     Status: Partial

  Scc Graph
     Nodes:  28501
     Arcs:   138718
     Secs:   1
```

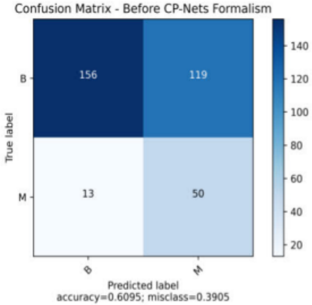
```
 Home Properties
------------------------------------------------

  Home Markings
     None


 Liveness Properties
------------------------------------------------

  Dead Markings
     16964 [28501,28500,28499,28498,28497,...]

  Dead Transition Instances
     Diagnosis_B_Labeller'Rule2 1
     Diagnosis_B_Labeller'Rule6 1
     Diagnosis_M_Labeller'Rule8 1

  Live Transition Instances
     None


 Fairness Properties
------------------------------------------------
     No infinite occurrence sequences.
```

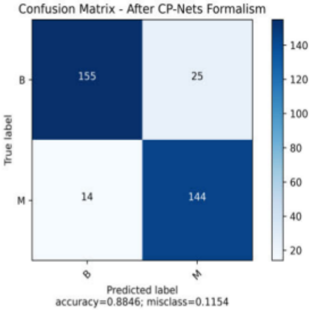
Recap


Data set
Test Data Set
Training Data Set
CPN Tokens
Training Data Set
Decision Trees
Supervised Machine Learning
Training Data Set
CPN Token
Testing Data Set
CPN Token
Decision Rule Extraction
Simulation & State
Space Analysis
Formally Verified
Prognostic Rules
Prognostic Rules
Initial CP-Nets
Revised CP-Nets
Formal Verification of Medical Diagnostic Prognostic Rules

Confusion matrix







Performance metrics

- Accuracy (ACC)

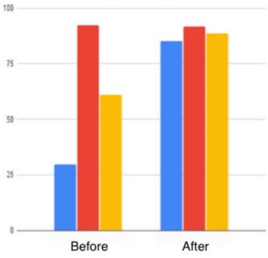
$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

- Sensitivity (TPR)

$$TPR = \frac{TP}{TP + FN}$$

- Specificity (TNR)

$$TNR = \frac{TN}{TN + FP}$$



Comparison with other methods

	Breast Cancer Wisconsin Diagnostics (WDBC)		
Method	Sensitivity	Specificity	Accuracy
ANN	97.25	97.31	97.14
SVM	95.98	94.44	96.82
K-NN	95.21	95.86	94.02
RF	95.73	97.84	92.64
NB	92.39	93.51	90.47
Proposed approach	91.71	85.20	88.46

Figure: Comparison with other supervised ML methods

	Breast Cancer Wisconsin Diagnostics (WDBC)			
Method	Rules	Sensitivity	Specificity	Accuracy
Decision Tables	27	93.75	89.59	96.91
OneR	1	88.5	87.28	89.42
PART	6	91.75	89.59	93.39
RIPPER	3	92.5	91.32	93.39
Proposed approach	6	91.71	85.20	88.46

Figure: Comparison with other rule extraction methods

References

[1] G. Geeraerts. *An Introduction to Petri nets and how to analyse them...* URL: https://verif.ulb.ac.be//ggeeraer/Tutorial-Perti-Nets-Geeraerts.pdf.

[2] K. Jensen. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*. Vol. 1. Berlin: Springer-Verlag, 1992.

[3] K. Jensen and Lars M. Kristensen. *Coloured Petri Nets: Modelling and Validation of Concurrent Systems*. Berlin: Springer-Verlag, 2009.

[4] M. Nauman et al. "Guaranteeing Correctness of Machine Learning Based Decision Making at Higher Educational Institutions". In: *IEEE Access* 9 (2021), pp. 92864–92880. DOI: 10.1109/ACCESS.2021.3088901.

[5] M. Nauman et al. "Improving the Correctness of Medical Diagnostics Based on Machine Learning With Coloured Petri Nets". In: *IEEE Access* 9 (2021), pp. 143434–143447. DOI: 10.1109/ACCESS.2021.3121092.