Prädiktion und Klassifikation mit Random Forest

Prof. Dr. T. Nouri
Nouri@nouri.ch

20.11.14

Übersicht

1. Probleme mit Decision Tree
2. Der Random Forests RF
3. Implementation & Tests RF
4. Resultate
5. Rückblick

Decision Tree: Supervised Classification

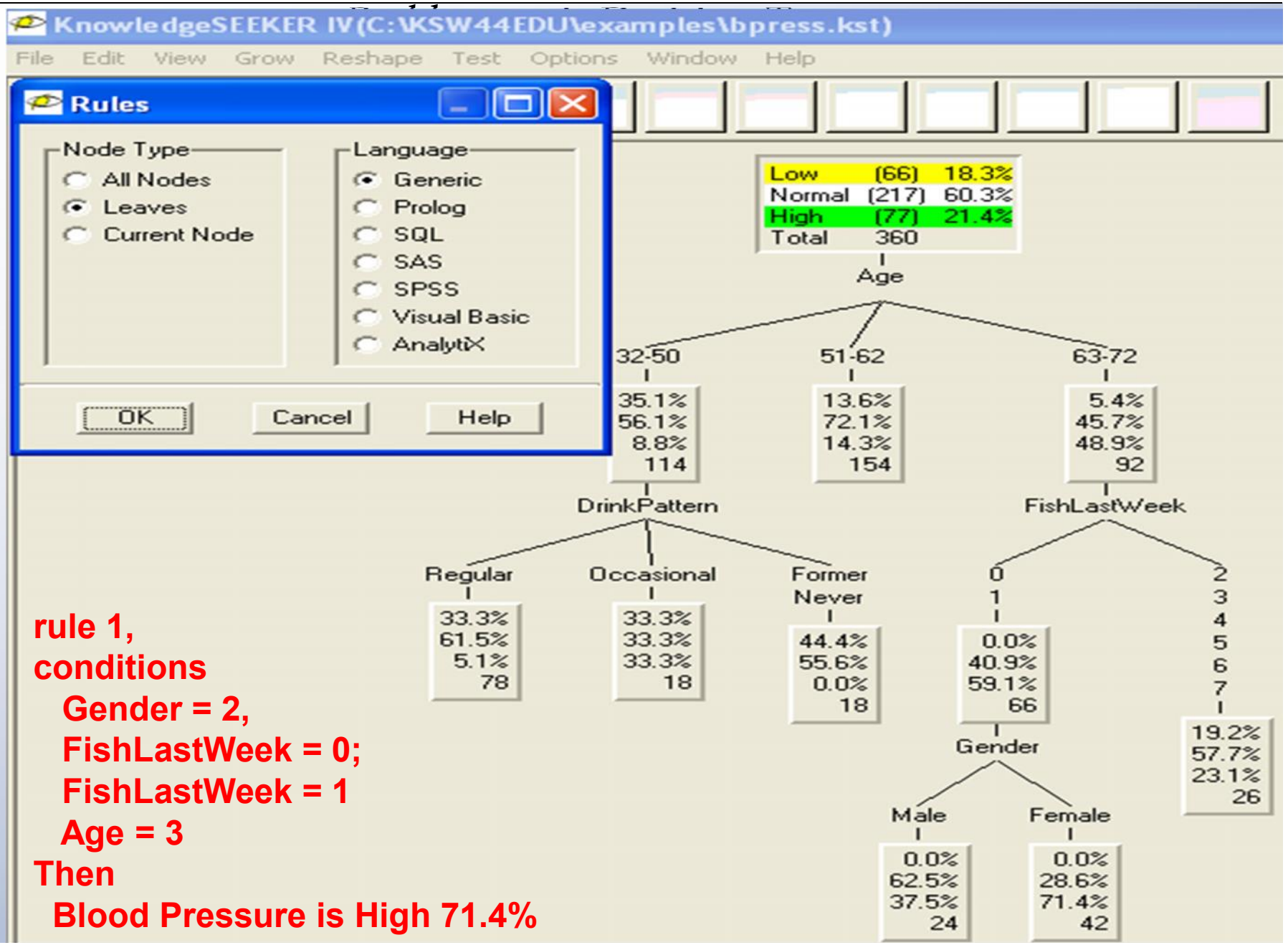
KnowledgeSEEKER IV(C:\KSW44EDU\examples\bpress.kst)

File Edit View Grow Reshape Test Options Window Help

View Data

View

	Pattern	DrinksEveryDay	Age	YearsEducation	Income	Gender	Weight	Height	zscore	Hypertension
1		2	3	2	16	1	141	626	4.530000	2
2		2	2	2	9	1	187	635	5.030000	3
3		1	2	2	11	1	191	707	3.840000	2
4		9	2	2	3	1	160	657	4.430000	2
5		2	2	2	15	1	216	709	4.430000	2
6		2	1	2	12	1	206	720	2.760000	1
7		9	3	2	9	2	215	632	4.900000	3
8		9	3	5	10	2	89	641	4.510000	2
9		2	1	2	11	1	188	679	3.750000	2
10		9	1	2	16	2	284	677	3.680000	2
11		9	3	5	10	2	150	626	4.370000	2
12		2	2	1	13	1	180	649	4.150000	2
13		2	1	2	7	2	167	619	4.320000	2
14		2	1	2	16	1	206	714	3.480000	2
15		1	1	2	9	1	182	746	4.190000	2



rule 1,
conditions
Gender = 2,
FishLastWeek = 0;
FishLastWeek = 1
Age = 3
Then
Blood Pressure is High 71.4%

Der Random Forests

1. Allgemeines

2. Die Erstellung eines Baumes
    1. „Random Training sets“ generieren
    2. Baum erstellen
    3. Wald testen

3. Scoring

1 Allgemeines zum Random Forest

- Erweiterung des traditionellen Entscheidungsbaum-Modells
- Viele Bäume
- Weniger Splitvariablen
- Zufällige Datensets
- Kein Baum ist gleich wie der andere

2.1 „Random Trainingsets“ generieren

1. Daten einlesen
2. Anzahl Objekte (ntrain) und Anzahl Variablen (mdim) berechnen
3. Zufällig ntrain Objekte aus allen Objekte auswählen (Wiederholungen sind erlaubt)
4. Immer ntrain Objekte ergeben ein Training-Set Trn_n
5. Mit jedem Training-Set wird ein Baum gebaut

2.1 „Random Trainingsets“ generieren

	Obj. Nbr	Class	Var_1	Var_2	Var_3	Var_4	Var_5
2x	1	2	2	1	1	0	63
	2	1	1	2	0	2	51
	3	1	1	0	0	1	32
	4	1	0	5	0	0	63
2x	5	1	2	4	0	2	63
	6	1	0	3	0	0	32
3x	7	2	6	1	0	0	51
3x	8	2	4	1	0	1	63
	9	2	1	1	2	0	51
	10	2	0	0	1	0	63

ntrain = Anzahl Objekte

Hier: ntrain = 10

(Die Trainingsdaten sollten repräsentativ für alle möglichen Daten sein.)

Wähle zufällig ntrain Objekte aus den InputDaten (Wiederholungen sind erlaubt)

→ TrainingSet Trn_1

→ TrainingSet Trn_2

2.2 Baum erstellen

1. Zufällig mtry Variablen aus den mdim Variablen aussuchen.
   mtry wird mit den Startparametern angegeben.
   Als gute Zahl für mtry wird empfohlen.
   Hier wird mtry = 2 verwendet.

$$mtry = \sqrt{mdim}$$

Var_1	Var_2	Var_3	Var_4	Var_5

2. Für die mtry Variablen wird berechnet, welche von ihnen die Trn_i -Daten am besten aufteilt. (find best split)
   Dies kann mit dem Gini-Index, Entropie, usw. berechnet werden.

   In diesem Fall wurde folgende Entropieformel verwendet:

$$\text{"Entropie"} = -p_+ \ln\left(p_+/n\right) - p_- \ln\left(p_-/n\right)$$

p_+ = Anz. richtig klassifizierte
p_- = Anz. falsch klassifizierte
n = Anzahl Objekte

2.2.1 Node 1

Obj. Nbr	Class	Var_1	Var_2	Var_3	Var_4	Var_5
2	1	1	2	0	2	51
4	1	0	5	0	0	63
6	1	0	3	0	0	32
7	2	6	1	0	0	51
7	2	6	1	0	0	51
7	2	6	1	0	0	51
8	2	4	1	0	1	63
8	2	4	1	0	1	63
8	2	4	1	0	1	63
9	2	1	1	2	0	51
Ent		3.25			6.11	

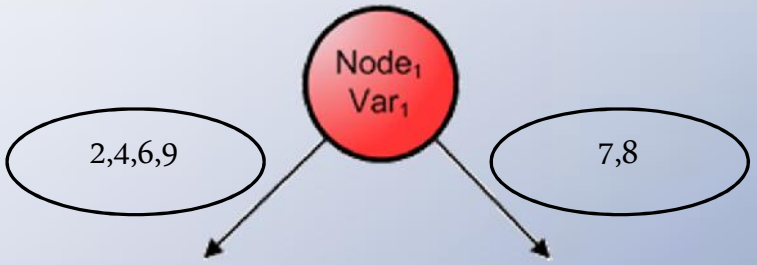
Im Node 1 wird der Best Split bei Var_1 und Var_4 gesucht.

Var_1 hat den kleineren „Entropie“-Wert

Var_1 ist der best Split.

Wenn Var 1 <= 2.5 dann Class = 1 ,

2.2.1 Node 1



Rule 1: Wenn Var 1 <= 2.5 dann Class = 1 , **Gehe zu Node 2**

Die Daten werden mit der Rule 1 aufgeteilt.

2.2.2 Node 2

Obj. Nbr	Class	Var_1	Var_2	Var_3	Var_4	Var_5
2	1	1	2	0	2	51
4	1	0	5	0	0	63
6	1	0	3	0	0	32
9	2	1	1	2	0	51
Ent		2.25	0			

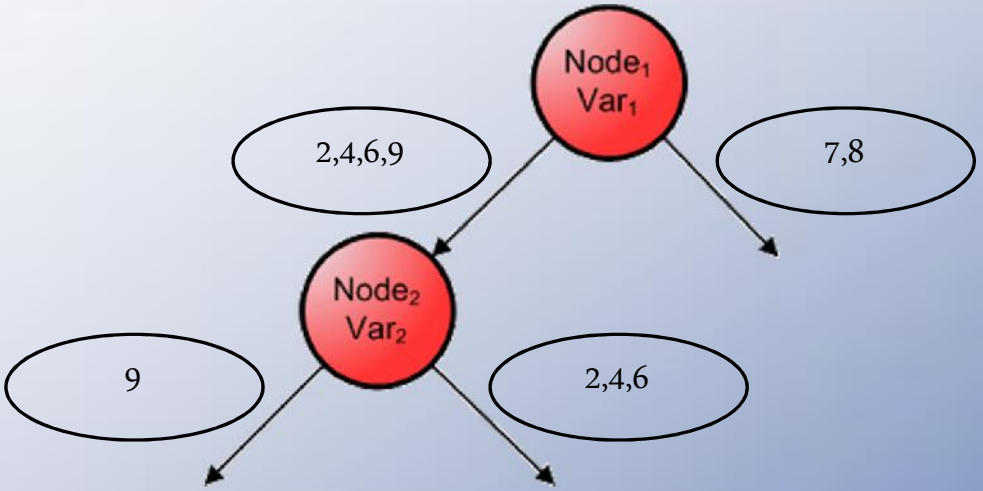
Im Node 2 wird der Best Split bei Var_1 und Var_2 gesucht.

Var_2 hat den kleineren „Entropie“-Wert

Var_2 ist der best Split.

Wenn Var 2 <= 1.5 dann Class = 2,

2.2.2 Node 2



Rule 2: Wenn Var 2 <= 1.5 dann Class = 2, **Gehe zu Node 4**

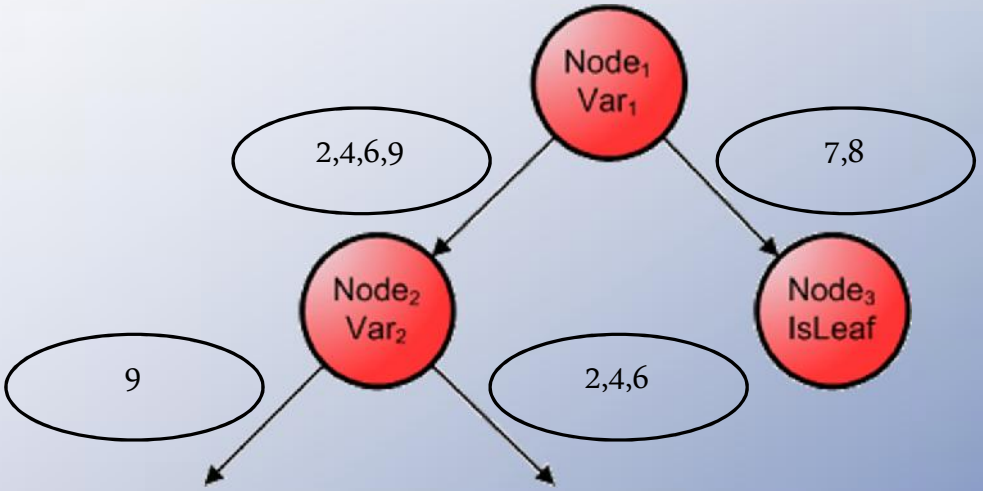
Die Daten werden mit der Rule 2 aufgeteilt.

2.2.3 Node 3

Obj. Nbr	Class	Var_1	Var_2	Var_3	Var_4	Var_5
7	2	6	1	0	0	51
7	2	6	1	0	0	51
7	2	6	1	0	0	51
8	2	4	1	0	1	63
8	2	4	1	0	1	63
8	2	4	1	0	1	63

Im Node 3 befinden sich nur noch Objekte mit Class = 2, der Knoten ist daher rein und muss nicht mehr weiter gesplittet werden.

2.2.3 Node 3

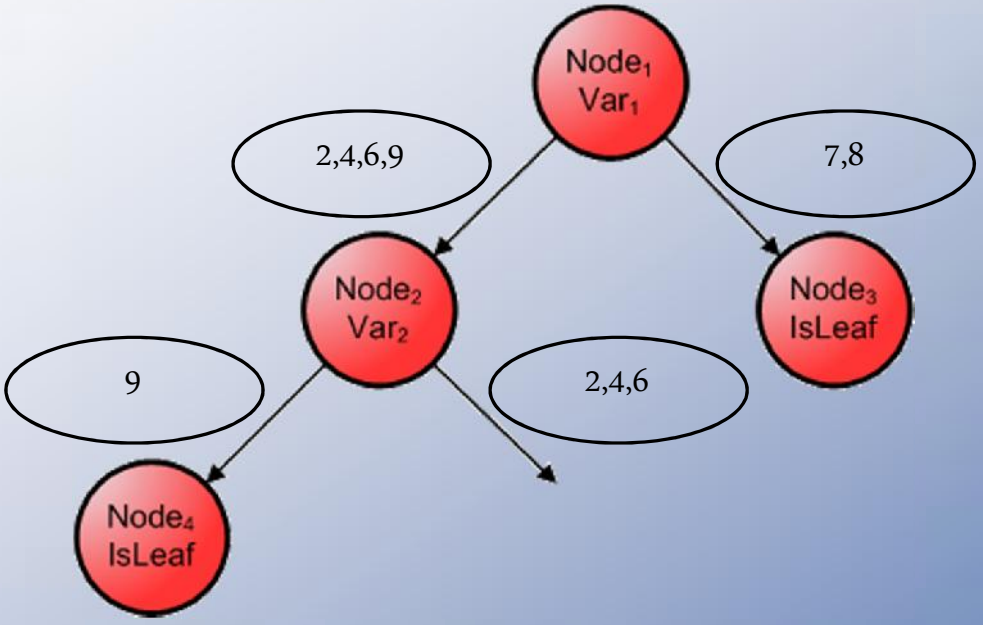


2.2.4 Node 4

Obj. Nbr	Class	Var_1	Var_2	Var_3	Var_4	Var_5
9	2	1	1	2	0	51

Im Node 5 befinden sich nur noch Objekte mit Class = 2, der Knoten ist daher rein und muss nicht mehr weiter gesplittet werden.

2.2.4 Node 4

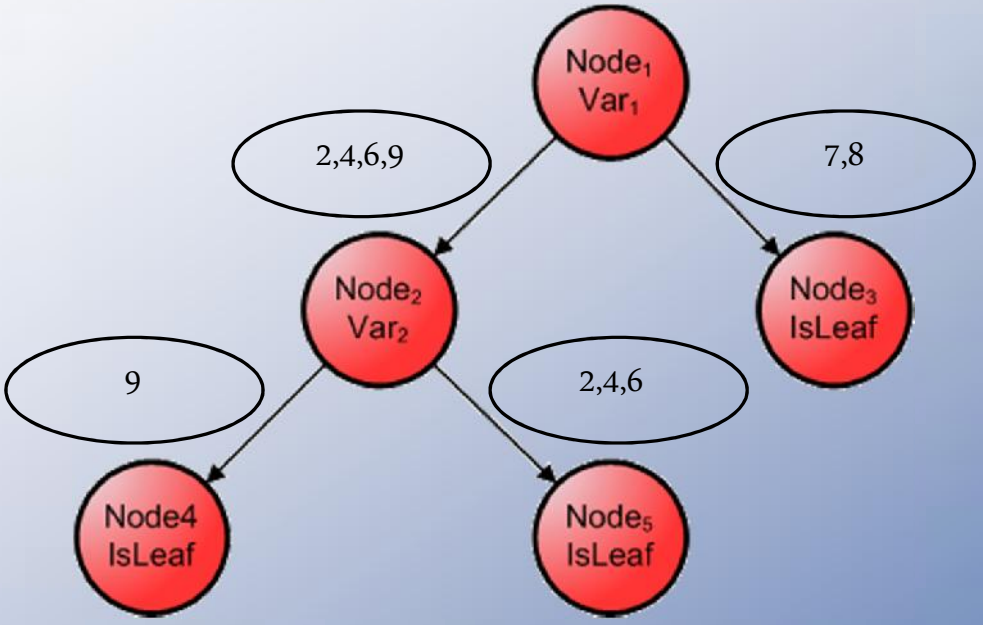

Node1
Var1
2,4,6,9
7,8
Node2
Var2
Node3
IsLeaf
9
2,4,6
Node4
IsLeaf

2.2.5 Node 5

Obj. Nbr	Class	Var_1	Var_2	Var_3	Var_4	Var_5
2	1	1	2	0	2	51
4	1	0	5	0	0	63
6	1	0	3	0	0	32

Im Node 4 befinden sich nur noch Objekte mit Class = 1, der Knoten ist daher rein und muss nicht mehr weiter gesplittet werden.

2.2.5 Node 5



Der Fertig entwickelte Baum

2.3 Wald evaluieren

- Oob-TestSet erstellen
- Objekte von Baum Klassifizieren lassen
- Fehlerquote berechnen

2.3.1 Oob-TestSet erstellen

Obj. Nbr	Class	Var_1	Var_2	Var_3	Var_4	Var_5
1	2	2	1	1	0	63
2	1	1	2	0	2	51
3	1	1	0	0	1	32
4	1	0	5	0	0	63
5	1	2	4	0	2	63
6	1	0	3	0	0	32
7	2	6	1	0	0	51
8	2	4	1	0	1	63
9	2	1	1	2	0	51
10	2	0	0	1	0	63

Für den Test wird für jeden Baum $Tree_i$ ein TestSet verwendet.
Dies sind jeweils diejenigen Objekte, die nicht im TrainingsSet Trn_i des Baums enthalten waren, gebildet.

2.3.2 Objekte Klassifizieren

Obj. Nbr	Class	Var_1	Var_2	Var_3	Var_4	Var_5
1	2	2	1	1	0	63

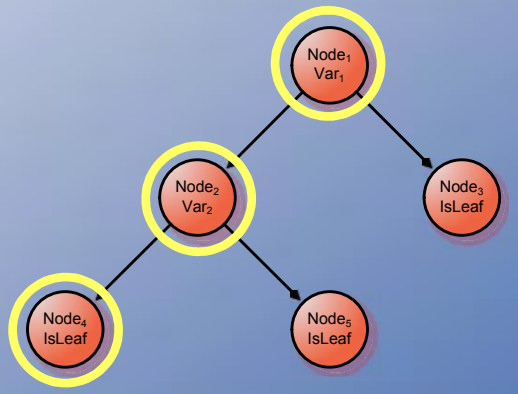
Rule 1: Wenn Var 1 <= 2.5 dann Class = 1

Gehe zu Node 2

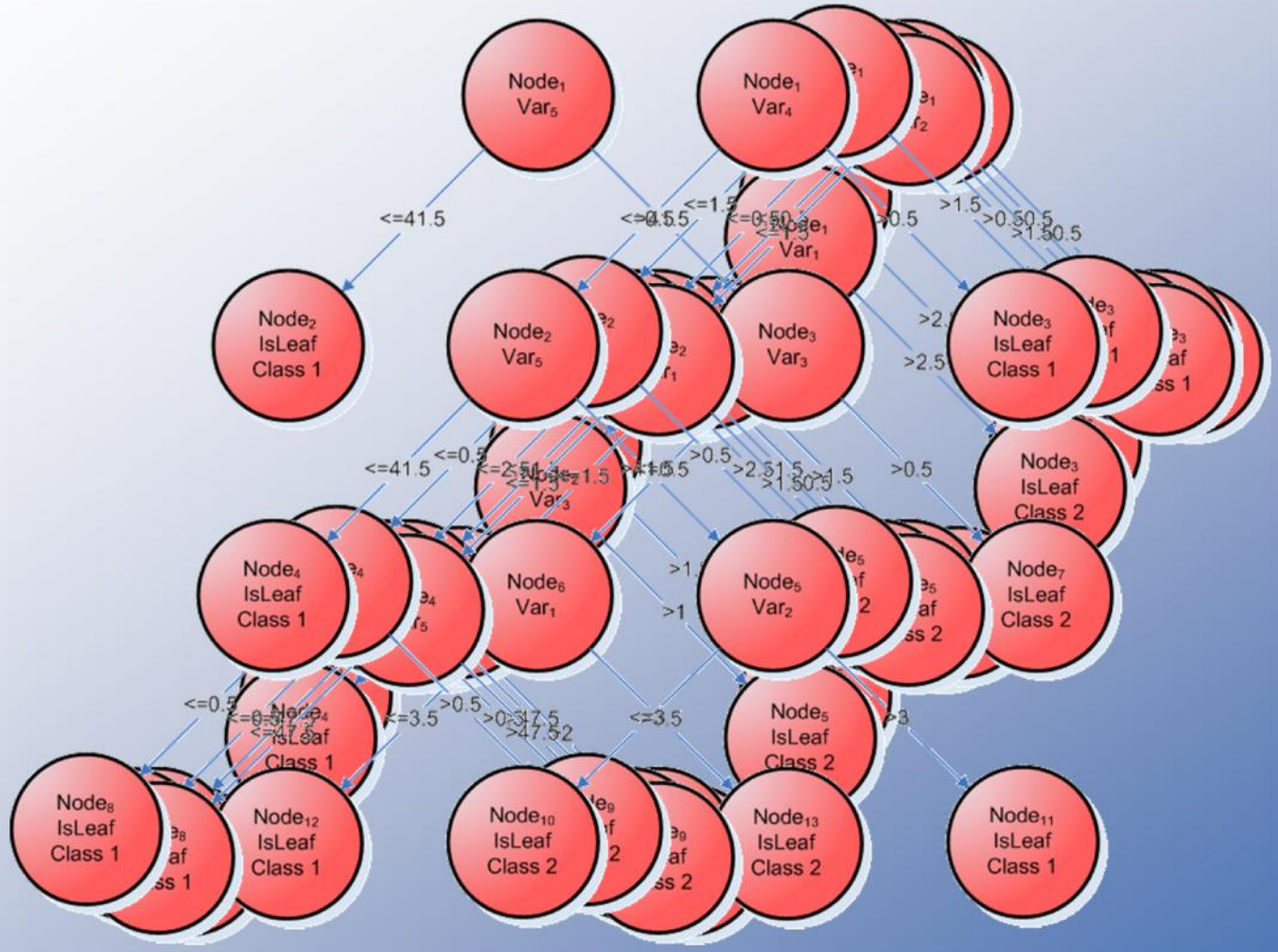
Rule 2: Wenn Var 2 <= 1.5 dann Class = 2

Gehe zu Node 4

Objekt1: Class = 2



2.3.3 Für alle Bäume ...



2.3.3 werden die Objekte Klassifiziert

- Jeder Baum klassifiziert sein Oob-Test Set

  $\Rightarrow$ Objekt Score

- Die Objekt Score gibt für jedes $Objekt_i$ an, mit welcher Wahrscheinlichkeit es falsch Klassifiziert wurde

- Aussagekraft des Waldes = Durchschnitt aller Objekt Scores

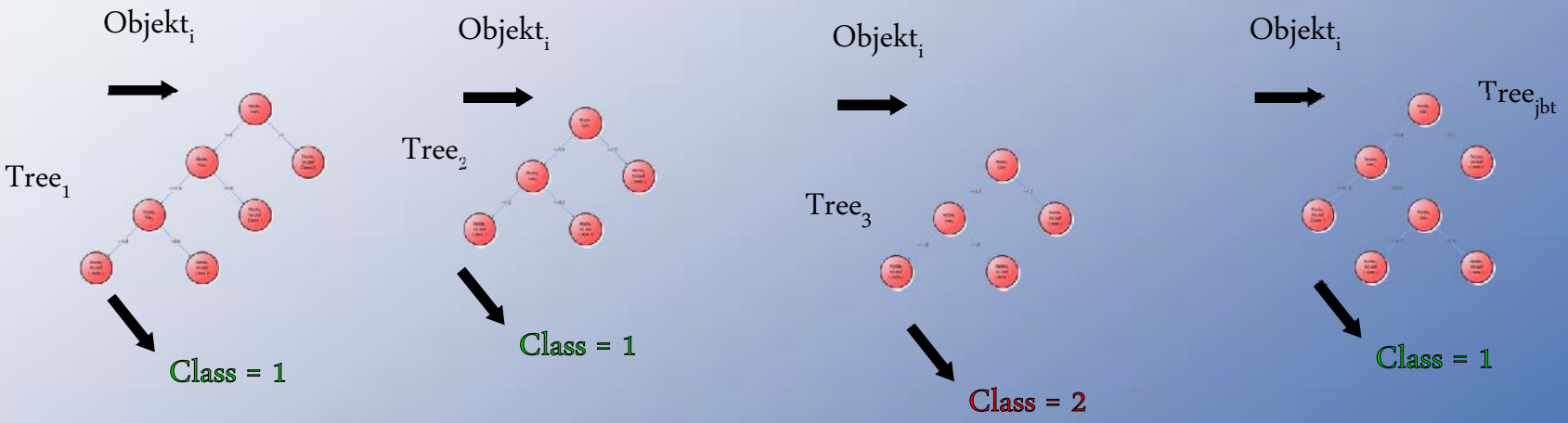
$$\text{Aussagekraft} = \frac{\sum_{i=1}^{mdim} \text{ObjektScore}_i}{mdim}$$

2.3.4 Fehlerquote berechnen

Objekt	Ausgewählt	Davon falsch	Fehler-%
1	4	1	25%
2	3	0	0%
3	2	1	50%
4	2	1	50%
5	3	2	66.67%
6	2	0	0%
7	6	3	50%
8	3	2	66.67%
9	1	1	100%
10	3	0	0%
		Total:	40.83%

3. Scoring

- Wenn ein Random Forest generiert wurde, können ihm beliebige Testdaten eingegeben werden.
- Jedes Test-Objekt wird von jedem Baum klassifiziert.



Der Test Fall wird somit derjenigen Klasse zugewiesen, die am Meisten Stimmen erhält:VOTING

b. Implementation & Tests von RF

- Implementierung des Modells in der SAS-Language
- Einbindung in den Enterprise Miner

c. Tests

- 4 Verschiedene Modelle
  - SAS Random Forest
  - Breiman Random Forest
  - Decision Tree 1
  - Decision Tree 2

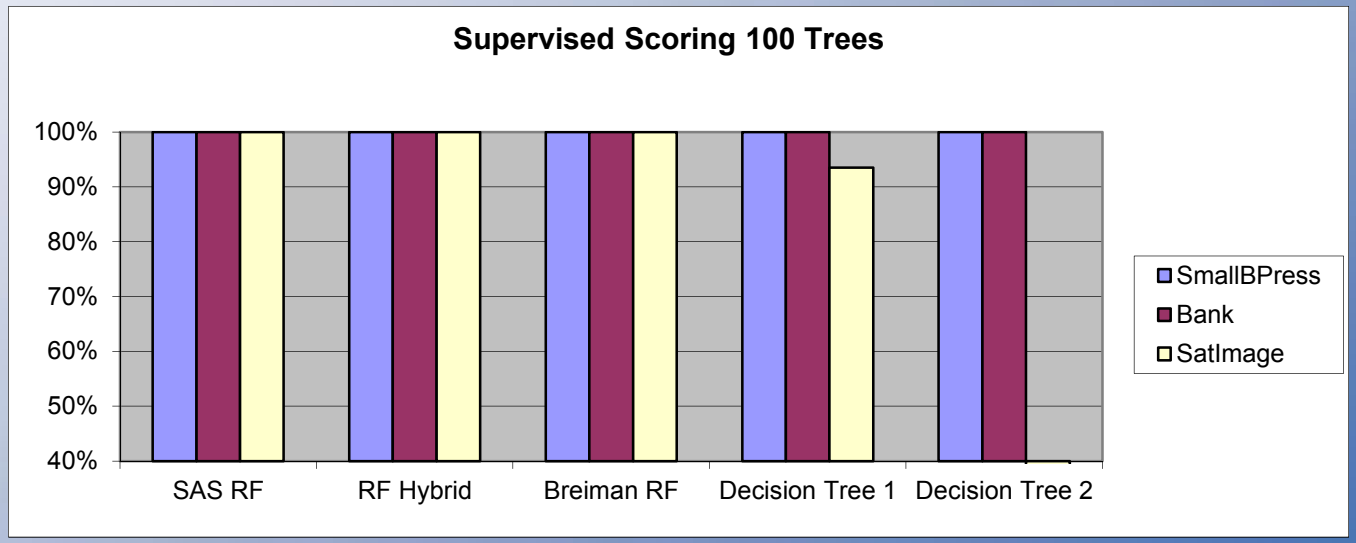
- 3 Verschiedene Datensets
  - SmallBPress 10 Training / 10 Test 5 Var.
  - Bank 50 Training / 500 Test 8 Var.
  - SatImage 4435 Training /2000 Test 36 Var.

c. Tests

- Durchläufe mit 1, 5, 10 und 100 Bäumen für alle Forest Modelle
- Supervised und Unsupervised Scoring mit allen Modellen

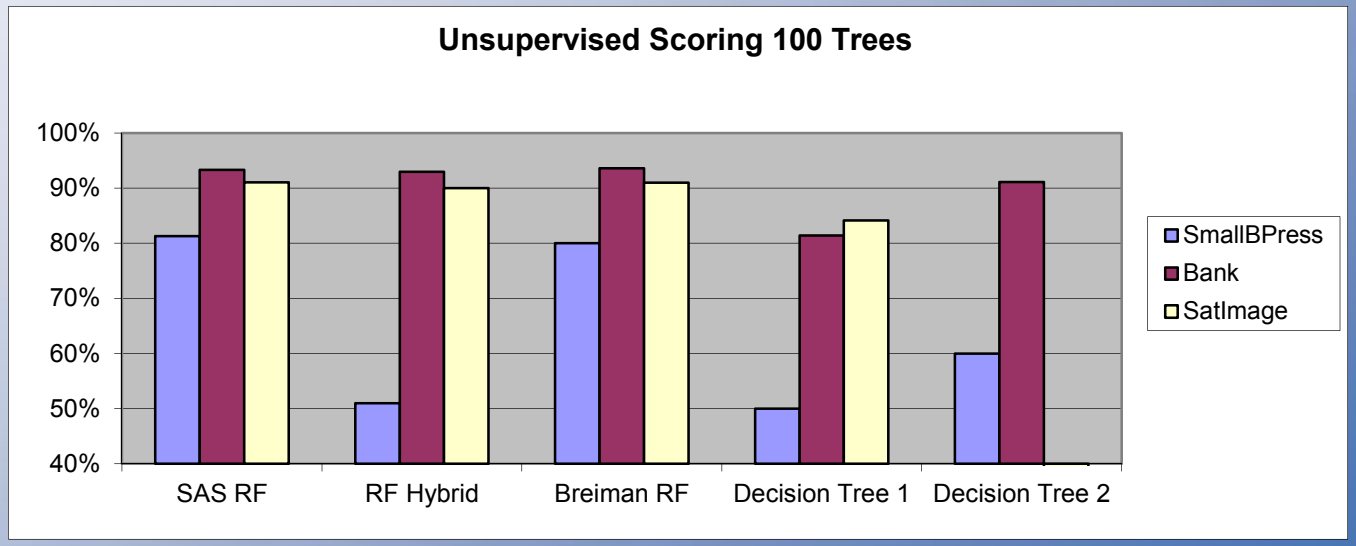
d. Resultate / Supervised

DatenSet	SAS RF	RF Hybrid	Breiman RF	Decision Tree 1	Decision Tree 2
SmallBPress	100%	100%	100%	100%	100%
Bank	100%	100%	100%	100%	100%
SatImage	100%	100%	100%	94%	-



d. Resultate / Unsupervised

DatenSet	SAS RF	RF Hybrid	Breiman RF	Decision Tree 1	Decision Tree 2
SmallBPress	81%	51%	80%	50%	60%
Bank	93%	93%	94%	81%	91%
SatImage	91%	90%	91%	84%	-



d. Resultate

- Random Forest liefert die besseren Resultate als traditionelle Modelle

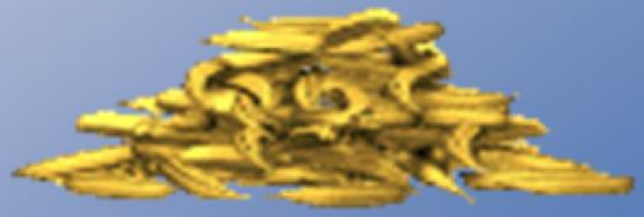
- Voraussetzung ist eine Mindestanzahl von Bäumen(mehr als 100)

- Somit ist das Random Forest - Modell eine bessere Alternative zum Decision Tree

Zusammenfassung





e. Rückblick

- Erste Kommerzielle Implementation

- Random Forest liefert bessere Ergebnisse Klassifikation/Vorhersage als Decision Trees

- Nur die besten aussagekräftigsten Bäume wählen ====> Besser Modell

Haben Sie noch Fragen?