

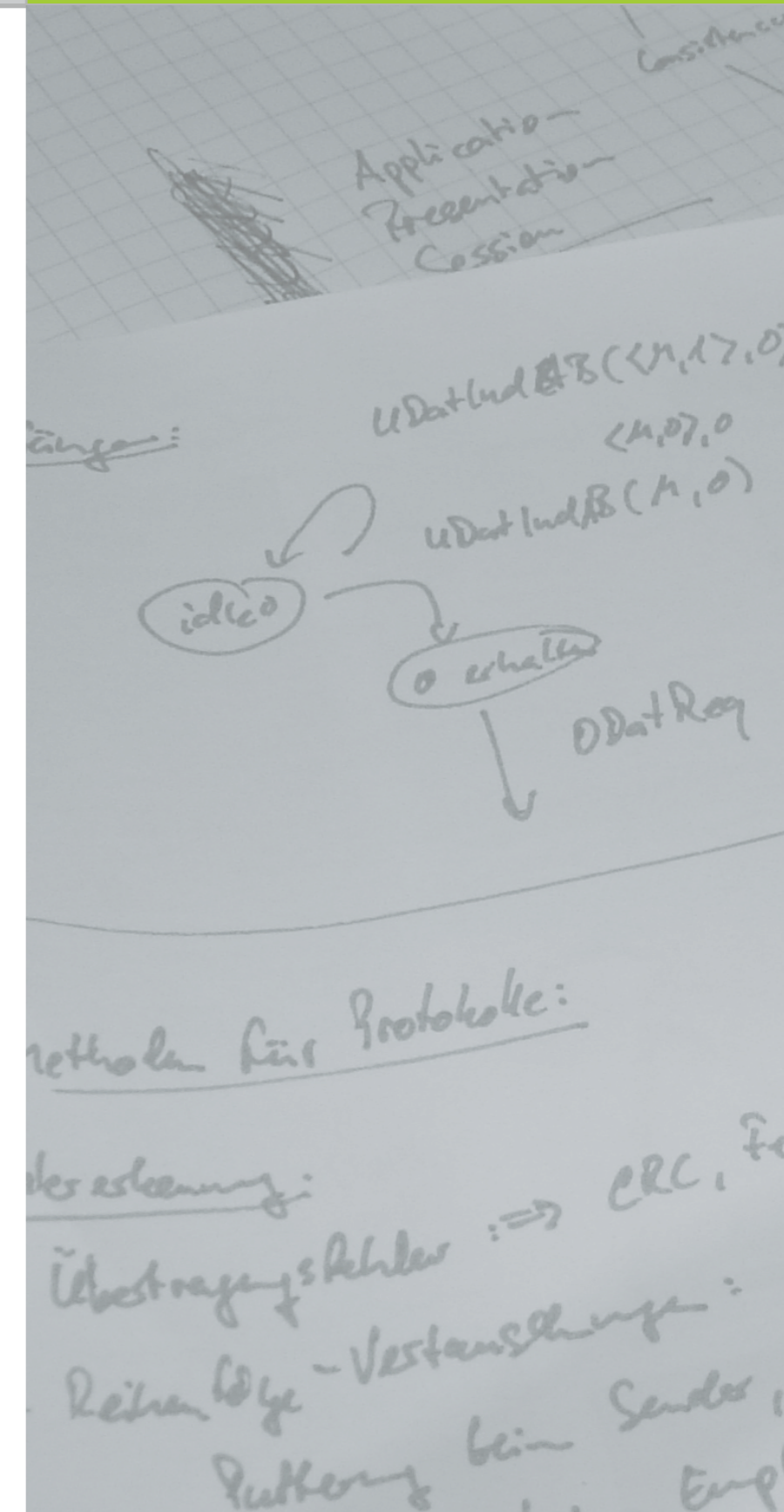
# Data Mining mit RapidMiner





# Motivation

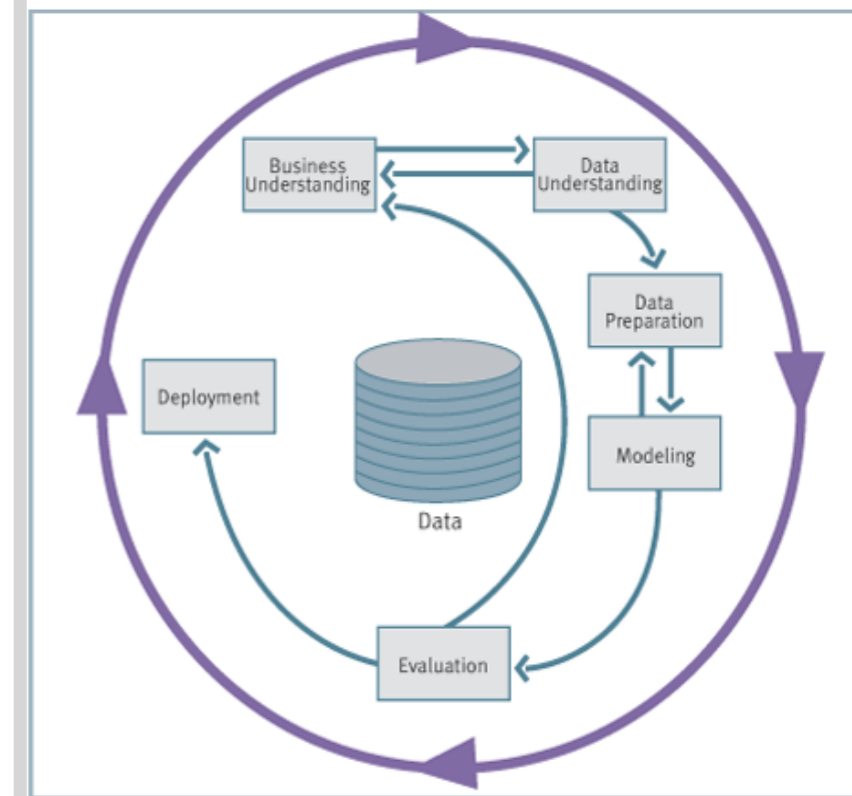
- CRISP: DM-Prozess besteht aus unterschiedlichen Teilaufgaben
- Datenvorverarbeitung spielt wichtige Rolle im DM-Prozess
- Systematische Evaluationen erfordern flexible und strukturierte Experimentierumgebung
- Ggf. periodische Wiederholungen von Analysen notwendig





# Anforderungen

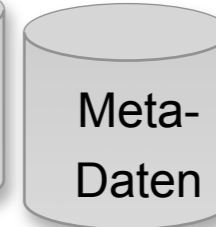
- Einfache wiederverwendbare Spezifikation von DM-Prozessen
- Austauschbarkeit von Lern-Verfahren, insbesondere:
  - Durchführen von Verfahrensvergleichen
- Kombination/Verschachtelung von Verfahren
- Verfahren zur Merkmalsauswahl und -generierung





# Konzept

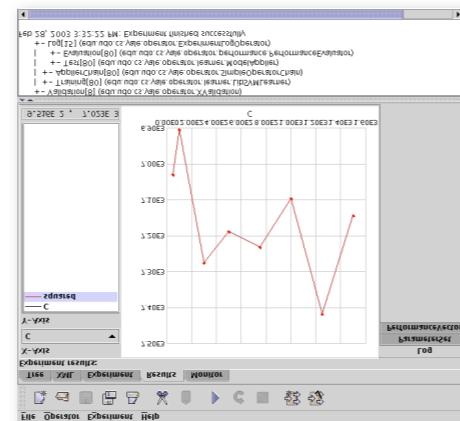
Data Mining  
Aufgaben



Entwurf

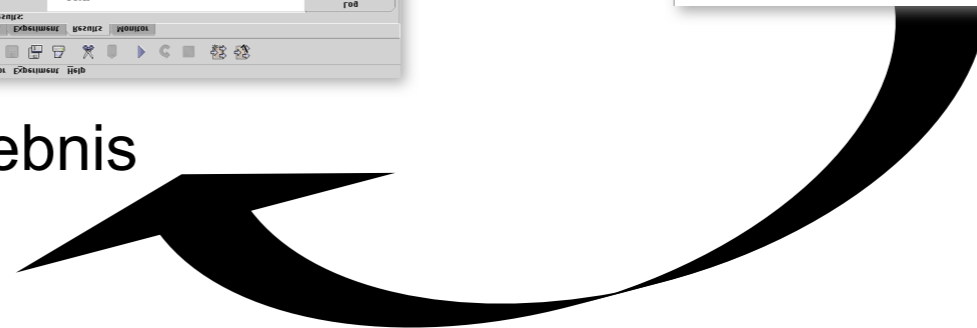
Data Mining Experiment

Durchführung



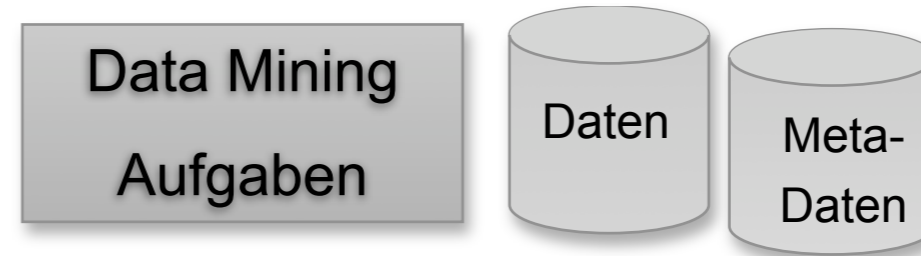
Key	Value
remove_unused	<input checked="" type="checkbox"/>
selection_direction	backward
keep_best	1
generations_without_improval	1

Ergebnis





# Konzept

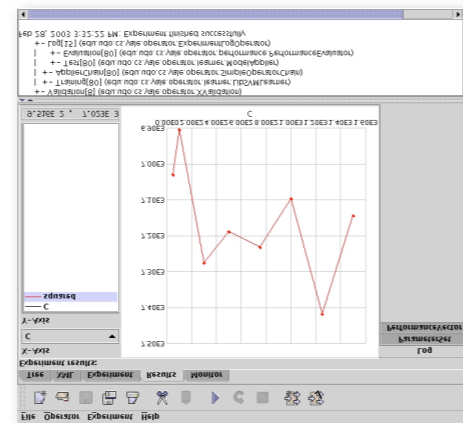


Entwurf

Anpassung des Entwurfes

Data Mining Experiment

Durchführung



Key	Value
remove_unused	<input checked="" type="checkbox"/>
selection_direction	backward
keep_best	1
generations_without_improval	1

Ergebnis



# Konzept

Data Mining  
Aufgaben

Daten

Meta-  
Daten

Entwurf

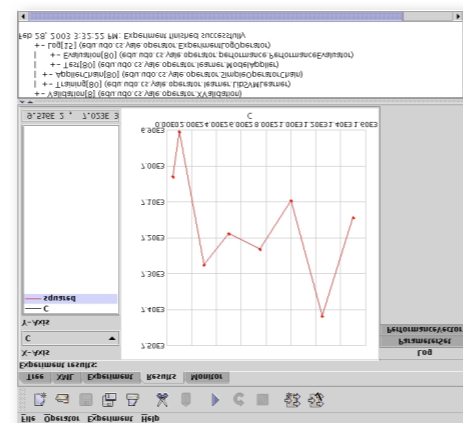
Anpassung des  
Entwurfes

Data Mining Experiment

Durchführung

## Data-Mining Experiment

Ergebnis



The figure shows a screenshot of a Data Mining Experiment software interface. It includes a tree view on the left showing a hierarchy of operators: FS, SelectionChain, XVal, Training, ApplierChain, Applier, Evaluation, and Expl. The main area displays a results table with columns 'Key' and 'Value'. The log window at the bottom shows the execution details of the 'Expl' operator.

Key	Value
remove_unused	<input checked="" type="checkbox"/>
selection_direction	backward
keep_best	1
generations_without_improval	1

```
<?xml version="1.0" encoding="UTF-8" ?>
<operator name="Expl" class="ExperimentLog">
  <dist key="log">
    <parameter key="generation" value="operator FS value generation">
      <parameter key="performance" value="operator FS value performance">
    </parameter>
  </parameter>
</operator>
</operator>
```



# Data Mining Aufgaben

- Klassifikation/Regression
- Transduktion
- Merkmalsextraktion
- Merkmalsgenerierung
- Merkmalsselektion
- Concept Drift
- Zeitreihenanalyse
- Text-Mining



# RapidMiner

- Modellierung von DM-Prozessen als Abfolge von Operatoren (Ketten)
- Verschachtelung von Operatoren
- Transparente/effiziente Datenhaltung
- Leichte Erweiterbarkeit
- GUI-Modus/Batch-Modus
- Einbindung externer Programme (z.B. Weka, SVM-Implementierungen)





# Integrierte Operatoren

- Operatoren zur Ein-/Ausgabe
- Datenvorverarbeitung
- Zahlreiche Lernverfahren  
(Weka-Lerner, Clustering, ...)
- Performanzbewertung von  
Lernverfahren
- Verwaltung/Ausgabe von  
Lernergebnissen



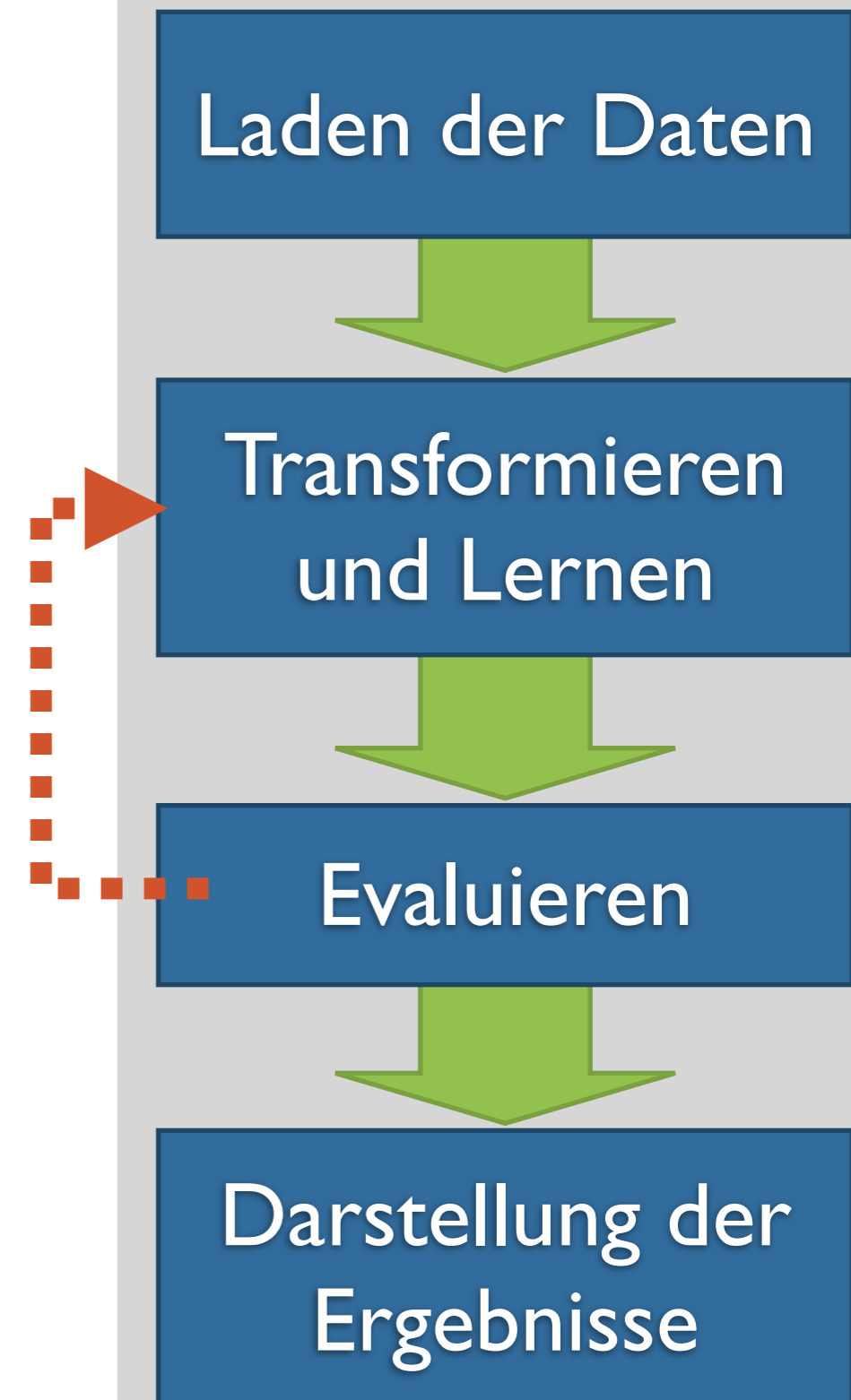
# Information

- Open-Source (GPL-Lizenz)
- Erfolgreiche Anwendung auf unterschiedliche Lernaufgaben
- Weltweite Verbreitung (Anwender / Wissenschaftler in über 30 Ländern)
- Dokumentation/Download/uvm unter <http://rapid-i.com>



# DM-Experiment

- Laden der Daten
  - Datenbank, Datei
- Transformieren und Lernen
  - Fehlende Werte? Normierung?  
Klassifikation? Clustering?
- Optimierung:
  - Verfahrensauswahl, Parameter
- Ausgabe der Ergebnisse
  - Performanz, Regeln, Cluster





# Operator/OperatorChain



# Operator/OperatorChain

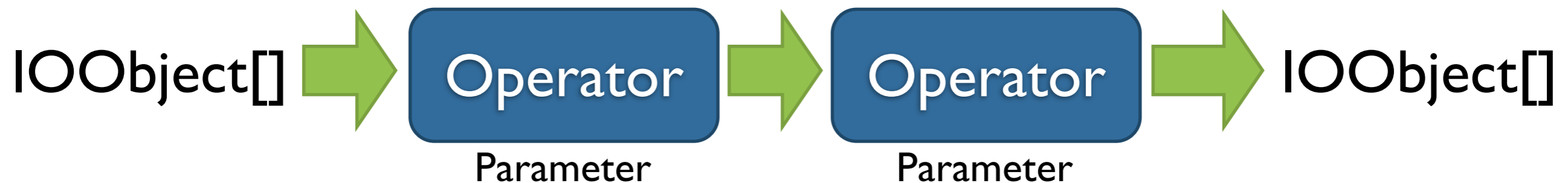
- Operator: Typ, Name, Eingabe, Ausgabe, Parameter





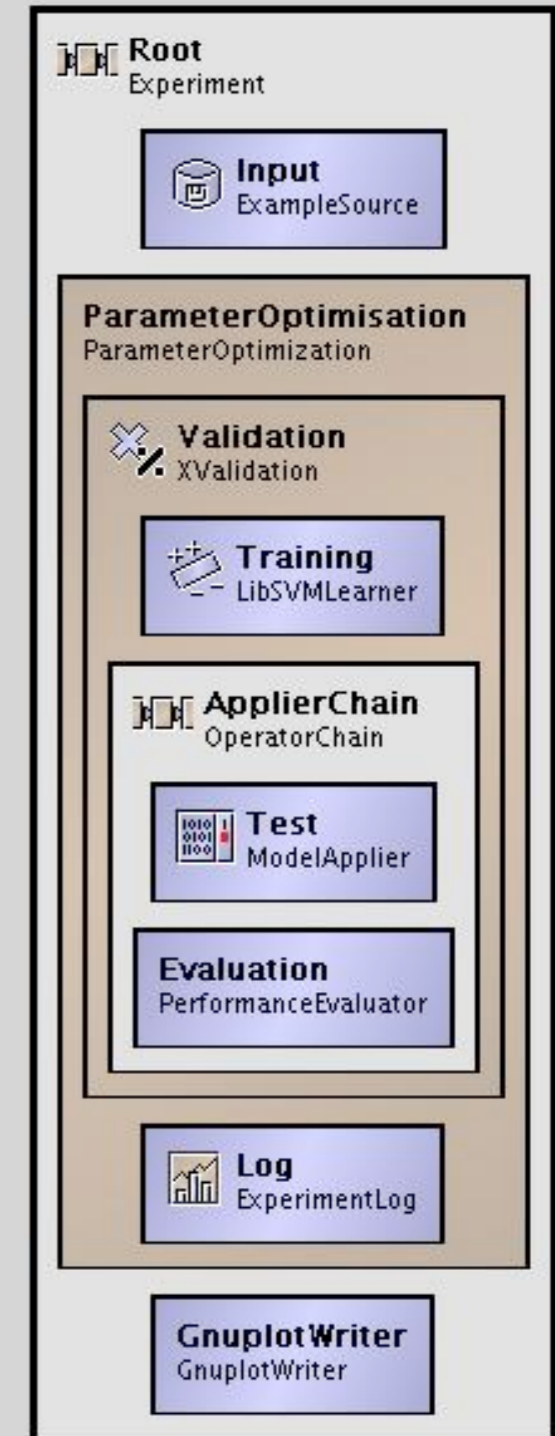
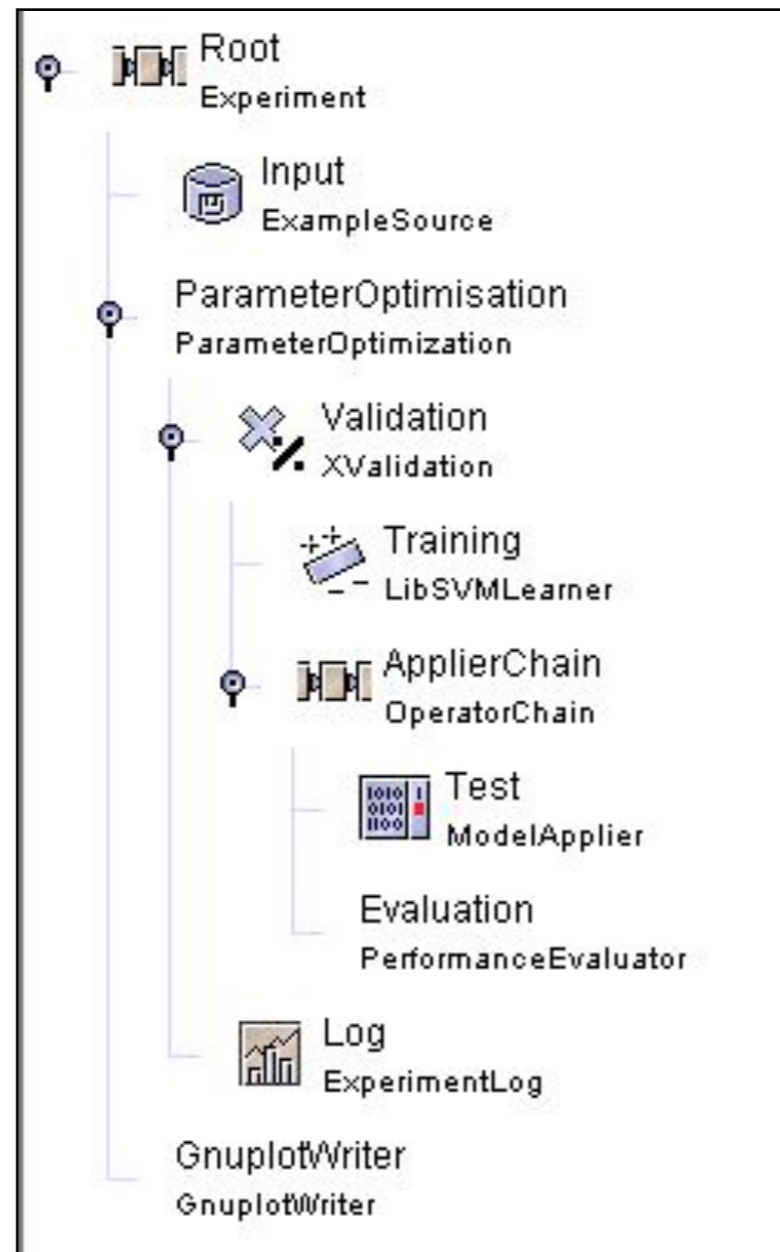
# Operator/OperatorChain

- Operator: Typ, Name, Eingabe, Ausgabe, Parameter





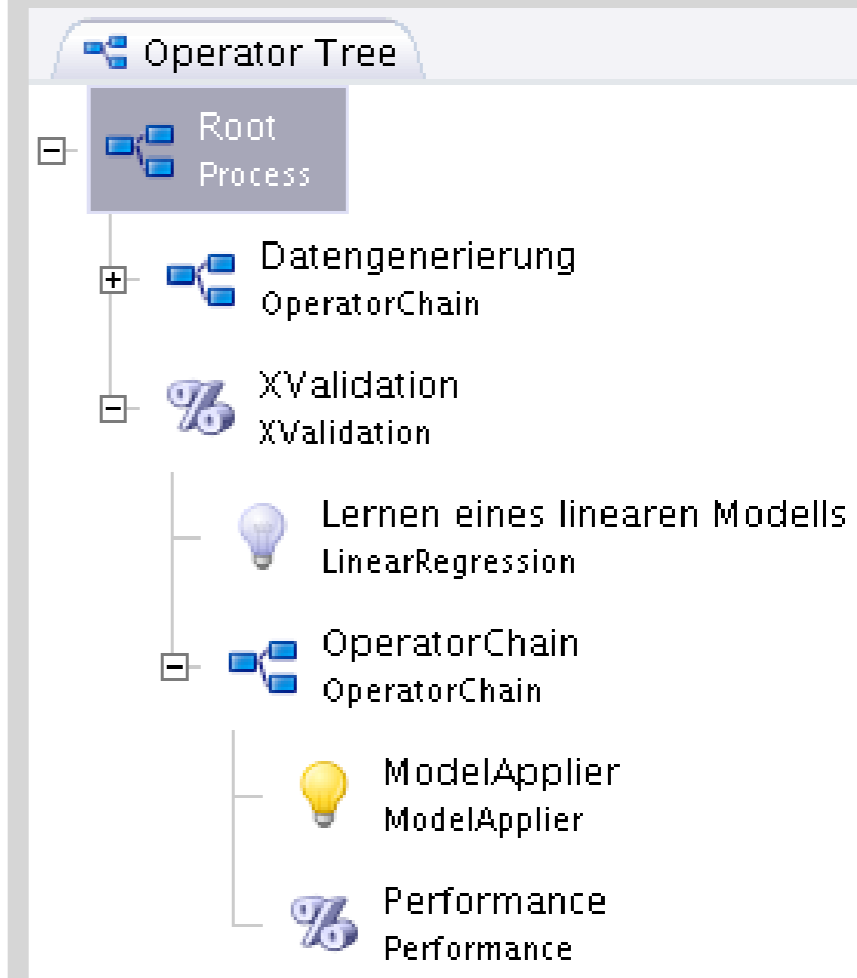
# Beispiel: Operatorbaum





# Beispiel: X-Validierung

- Zur Erinnerung:
  - Aufteilen der Daten in  $n$  Teilmengen
  - Lernen eines Modells auf  $n-1$  Mengen, Testen des Modells auf der übriggebliebenen Menge







# Parameter

## Aufbau/ Ablauf

The screenshot displays the RapidMiner interface with the following components:

- Operator Tree:** A hierarchical view on the left showing the process structure: Root Process -> Datengenerierung OperatorChain -> XValidation XValidation -> Lernen eines linearen Modells LinearRegression -> OperatorChain OperatorChain -> ModelApplier ModelApplier -> Performance Performance.
- Parameters Panel:** A table on the right showing configuration options for the selected operator. A red circle highlights this panel.
- Log Window:** A text-based log at the bottom showing the execution process. A red circle highlights this window.

Parameter	Value	Checkbox
keep_example_set		<input type="checkbox"/>
create_complete_model		<input type="checkbox"/>
average_performances_only		<input checked="" type="checkbox"/>
leave_one_out		<input type="checkbox"/>
number_of_validations	10	
sampling_type	stratified sampling	
local_random_seed	-1	

```
P Oct 15, 2008 3:41:42 PM: Initialising process setup
P Oct 15, 2008 3:41:42 PM: [NOTE] No filename given for result file, using stdout for logging results!
P Oct 15, 2008 3:41:42 PM: Checking properties...
P Oct 15, 2008 3:41:42 PM: Properties are ok.
P Oct 15, 2008 3:41:42 PM: Checking process setup...
P Oct 15, 2008 3:41:42 PM: Inner operators are ok.
P Oct 15, 2008 3:41:42 PM: Checking i/o classes...
P Oct 15, 2008 3:41:42 PM: i/o classes are ok. Process output: PerformanceVector.
P Oct 15, 2008 3:41:42 PM: Process ok.
P Oct 15, 2008 3:41:42 PM: Process initialised
P Oct 15, 2008 3:41:42 PM: [NOTE] Process starts
```

## Logfenster



# 100bject

- Objekte, die zwischen Operatoren ausgetauscht werden
- Beispiele:
  - ExampleSet (eine Menge von Daten)
  - Model (gelerntes Model)
  - PerformanceVector (Menge von Leistungsmaßen)
  - Ähnlichkeit
  - Merkmalsgewichte
  - ...



# ExampleSet (100bject)

- Beschreibung der Attribute (Metadaten):
  - Name
  - Skala: nominal, integer, real, ...
  - Einheit
  - Typ: Einzelwert, Zeitreihe, ...
  - Position (Spalte) in der Datendatei
- Sicht auf Daten



# ExampleSet (100bject)

- Spezielle Attribute:
  - Label
  - Predicted label
  - Id
  - Cluster
- Beliebig erweiterbar...



# ExampleSet (100bject)

File Table				
golf.data (1)	golf.data (2)	golf.data (3)	golf.data (4)	golf.data (5)
Outlook	Temperature	Humidity	Wind	Play
<b>attribute</b> ▼	<b>attribute</b> ▼	<b>attribute</b> ▼	<b>attribute</b> ▼	<b>label</b> ▼
[unit]	[unit]	[unit]	[unit]	[unit]
<b>nominal</b> ▼	<b>integer</b> ▼	<b>integer</b> ▼	<b>nominal</b> ▼	<b>nominal</b> ▼
<b>single...</b> ▼	<b>single...</b> ▼	<b>single...</b> ▼	<b>single...</b> ▼	<b>single...</b> ▼
sunny	85.0	85.0	false	no
sunny	80.0	90.0	true	no
overcast	83.0	78.0	false	yes
rain	70.0	96.0	false	yes
rain	68.0	80.0	false	yes
rain	65.0	70.0	true	no
overcast	64.0	65.0	true	yes
sunny	72.0	95.0	false	no
sunny	69.0	70.0	false	yes



# ExampleSet (IOObject)

- Input: -
- Output: ExampleSet
- Parameter: Attributdatei, Datendatei, Sampling, ...

Key	Value
<b>attributes</b>	data/golf.xml <b>Edit</b> ...
sample_size	-1
datamanagement	<b>double_array</b> ▼
separator_chars	::
ignore_chars	
comment_chars	#

Apr 26, 2004 11:15:45 AM: ParameterOptimisation returned additional output:



# 100objects/Resultate

YALE (simple.xml\*)

File Operator Experiment Help

Tree XML Experiment Results Monitor

Experiment results:

```

graph TD
    Wind[Wind] -- true --> Outlook1[Outlook]
    Wind -- false --> Temperature1[Temperature]
    Outlook1 -- rain --> no1[no]
    Outlook1 -- overcast --> yes1[yes]
    Outlook1 -- sunny --> Temperature2[Temperature]
    Temperature2 -- ≤75.0 --> yes2[yes]
    Temperature2 -- >75.0 --> no2[no]
    Temperature1 -- ≤83.0 --> yes3[yes]
    Temperature1 -- >83.0 --> no3[no]
    
```

Save...

Apr 26, 2004 10:42:10 AM: Root returned additional output:  
 1. Model (type Tree) for label #4: Play:=Play[]: nominal/single\_value/no block [yes,no]  
 Apr 26, 2004 10:42:10 AM: Root: execution time was 224 ms  
 Apr 26, 2004 10:42:10 AM: Experiment finished after 0 seconds  
 Apr 26, 2004 10:42:10 AM: Experiment:  
 Root[1] (Experiment)  
 +- Input[1] (ExampleSource)  
 +- NewOperator[1] (DecisionTreeLearner)

11:12:20 AM