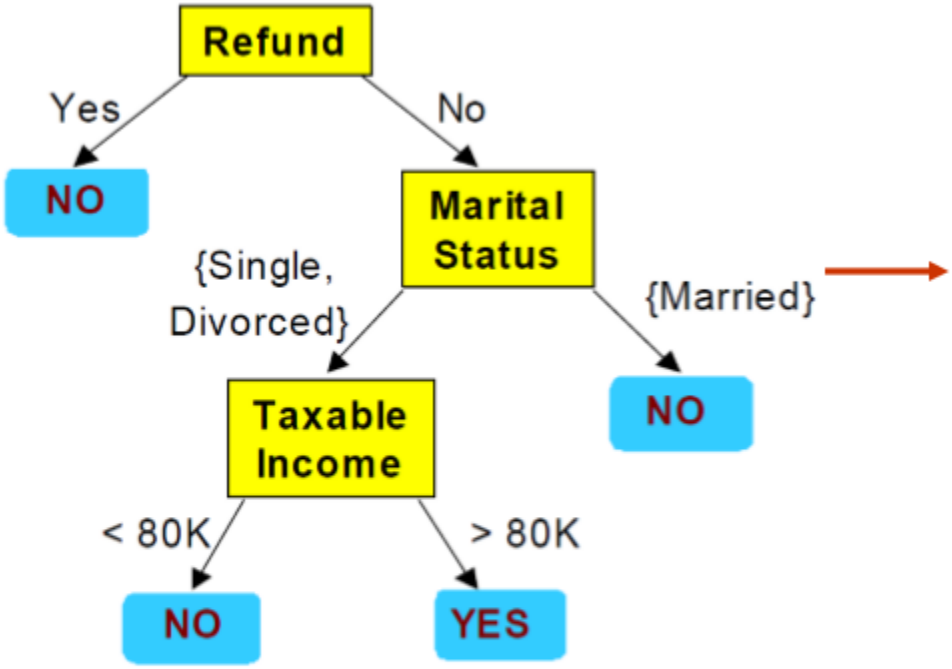
Classification

Exercise 4

Rule-based Classification



Classification Rules

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

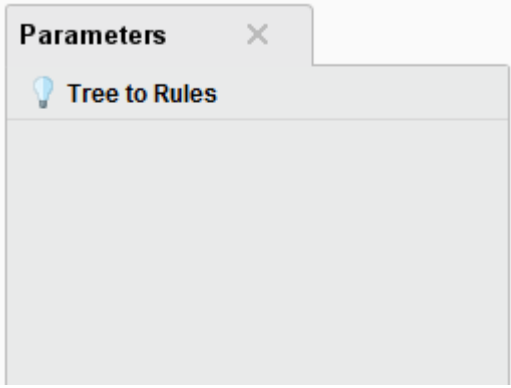
(Refund=No, Marital Status={Married}) ==> No

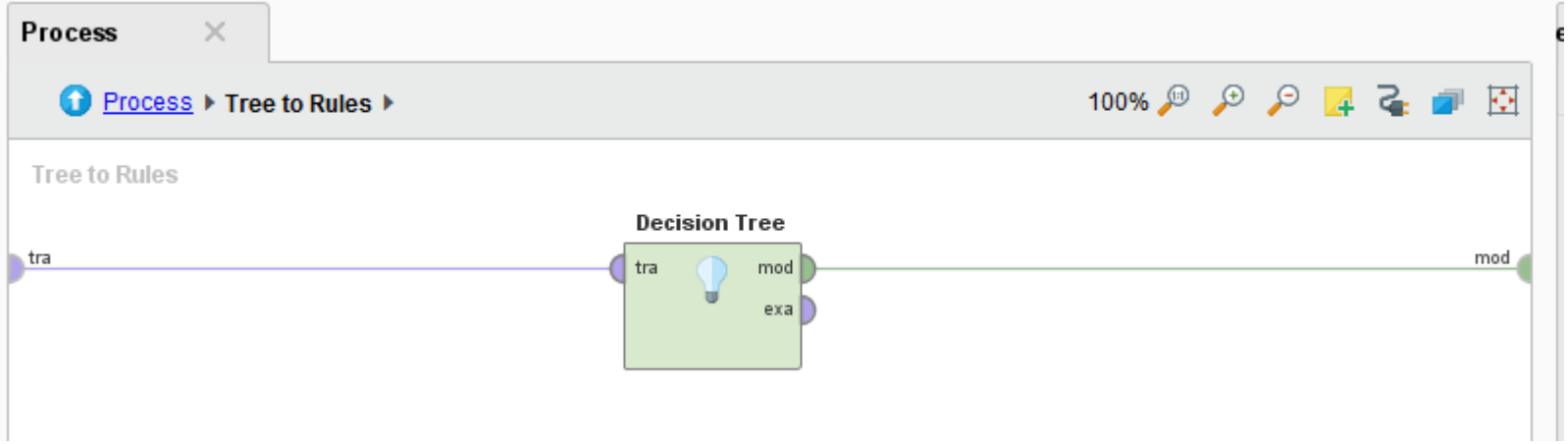
Rules are mutually exclusive and exhaustive

Rule set contains as much information as the tree

Operators: Tree to Rules

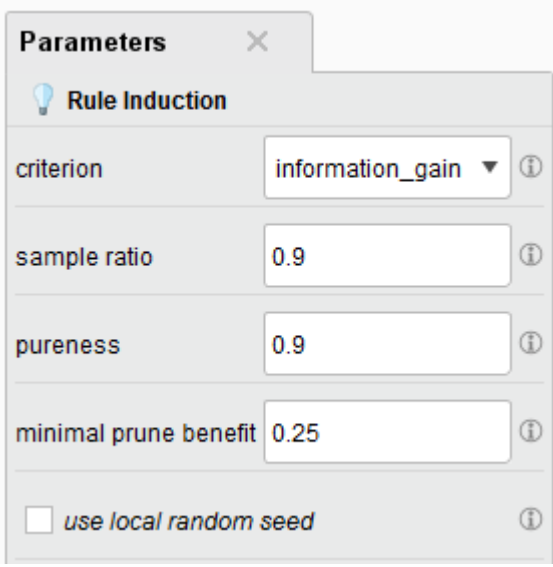
- Input Port:
  - Training data (Example Set)
- Output Ports:
  - Classification Model
  - Training data (Example Set)
- Parameters
  - None
- Has a nested process to learn a tree

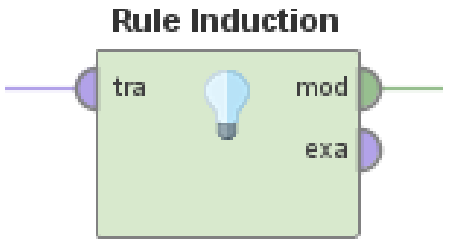




Operators: Rule Induction

- Input Ports
  - Training data (Example Set)
- Output Ports
  - Classification Model
  - Training data (Example Set)
- Parameters
  - Criterion for selecting split attribute
  - Sample ratio
  - Pureness (min.)
  - Minimal prune benefit





Rule Model Output

Examples of class 'good'

Examples of class 'bad'

RuleModel

```
if checking_status = no checking and other_payment_plans = none then good  (303 / 27)
if duration ≤ 17 and property_magnitude = real estate and age > 29.500 then good  ... / 6)
if duration ≤ 20.500 and credit_amount > 1285.500 and housing = own and job = ...resident then good  (23 / 4)
if duration ≤ 25 and credit_amount > 1373 and employment = 4<=X<7 and credi... then good  (32 / 4)
if credit_amount ≤ 3913.500 and own_telephone = yes and age > 35.500 the...
if checking_status = <0 and installment_commitment > 2.500 then bad  (43 /
if savings_status = no known savings and property_magnitude = car then good
if credit_amount ≤ 7391 and purpose = used car then good  (20 / 2)
if credit_amount ≤ 4086.500 and credit_amount > 2841 and duration ≤ 40.500 ...2.500 then good  (17 / 4)
if duration > 20.500 and age ≤ 28.500 and duration > 25.500 and duration ≤ 54 then bad  (4 / 24)
if age ≤ 32.500 and purpose = radio/tv and credit_amount ≤ 1268.500 then good  (17 / 1)
if age ≤ 32.500 and job = skilled and duration > 22.500 then good  (20 / 5)
if credit_amount > 11025 then bad  (3 / 15)
if personal_status = female div/dep/mar and checking_status = 0<=X<200 then bad  (13 / 28)
if age ≤ 31.500 and credit_amount > 1289.500 and property_magnitude = car and age > 22.500 then good  (15 / 0)
if property_magnitude = car and credit_amount ≤ 2519.500 then bad  (1 / 14)
if duration ≤ 16.500 and savings_status = <100 and installment_commitment > 1.500 then good  (20 / 2)
if savings_status = <100 and credit_amount ≤ 4256.500 then bad  (7 / 19)
if age ≤ 41.500 and age ≤ 28.500 and credit_amount > 1499 then good  (9 / 0)
if age > 41.500 and credit_amount > 2597 and duration ≤ 39 then bad  (0 / 9)
if personal_status = male single and savings_status = no known savings then good  (13 / 1)
if duration > 47.500 and age > 32 then bad  (0 / 8)
if credit_amount > 5152.500 and duration > 39 then good  (5 / 0)
if age ≤ 30.500 and credit_amount > 1135.500 then bad  (0 / 7)
else good  (14 / 9)

correct: 844 out of 996 training examples.
```

Balancing

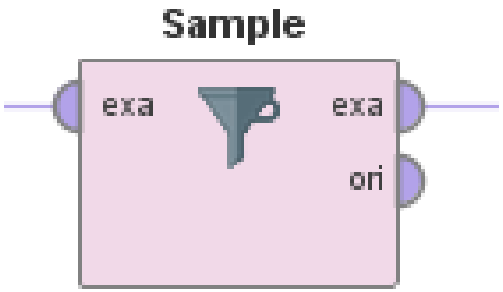
- The number of instances per class affects model learning and evaluation
  - Same as with normalisation, we don't want to emphasis one specific class
  - So, we make sure all classes are evenly distributed

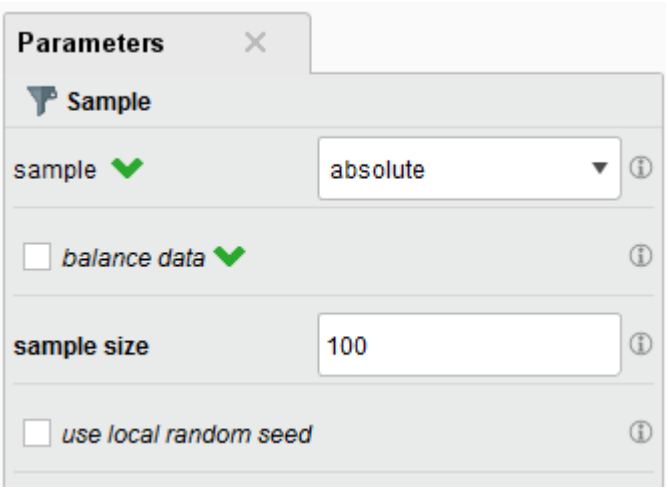
- Example: 9,999 examples for class A, 1 for class B
  - Classify everything as class A has 99.99% Accuracy
  - K-NN with $K \geq 3$ will always predict class A

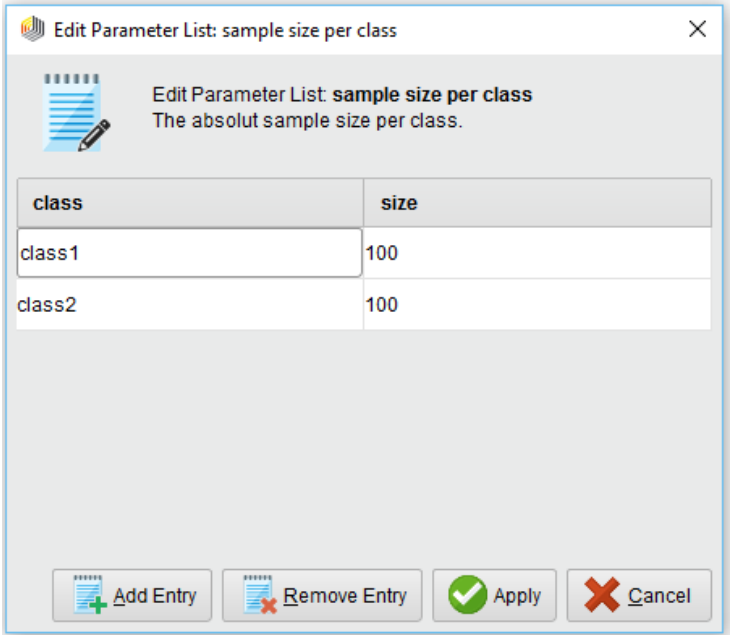
- Counter-Example: 500 examples for class A and B
  - Classify everything as class A has 50% Accuracy

Operators: Sample / Downsampling in Rapid Miner

- Input Port
  - Example Set
- Output Ports
  - Sampled data (Example Set)
  - Original Example Set
- Parameters
  - Sample (type)
  - Balance data (downsampling)
  - Sample size



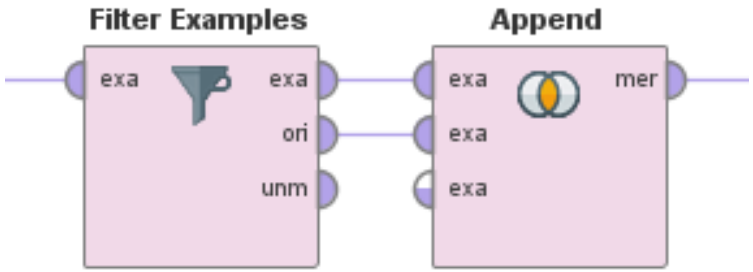




Upsampling in Rapid Miner

- Goal: Duplicate the examples of the under-represented class
  - Create a filter that only keeps the smaller class
  - Append the result to the original example set

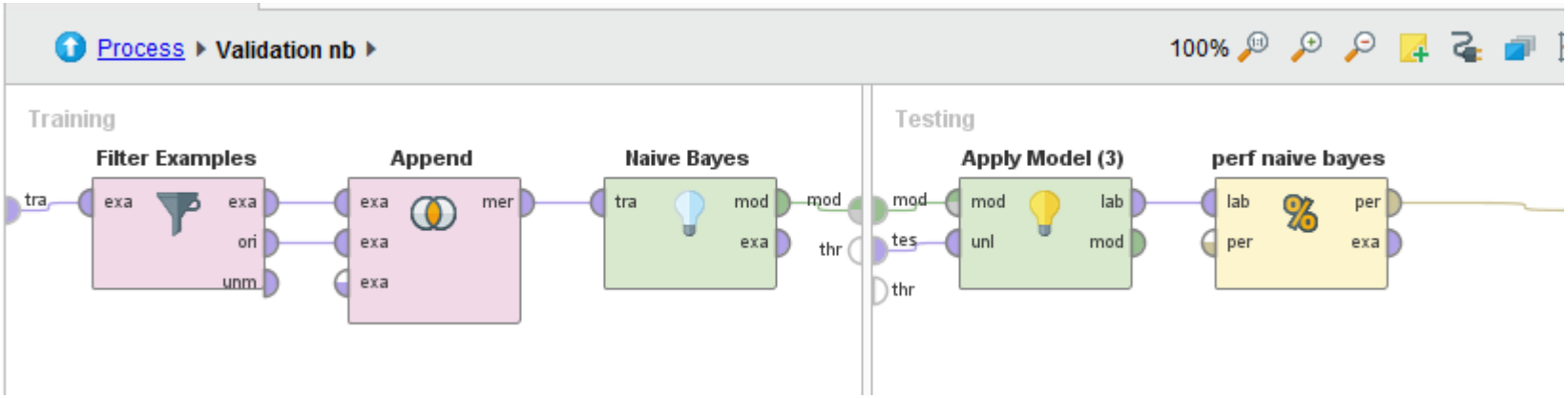
- What's the effect?
  - Better relative distribution of the classes (more balanced)
  - But: no new information from duplicated examples
  - Better: find new examples for the small classes



Balancing is only done for training data!!!

- Otherwise, you would evaluate on duplicate examples!

Never balance your test data!



Additional Evaluation Methods

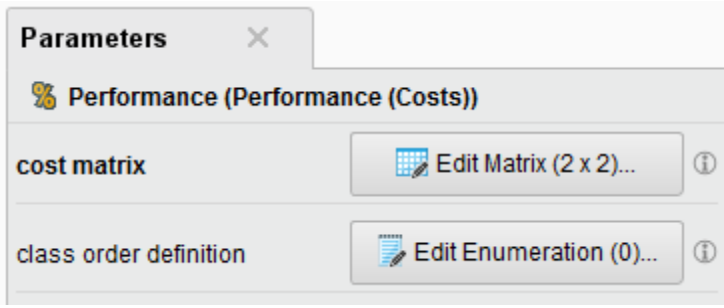
- Last exercise we looked at Accuracy, Precision & Recall

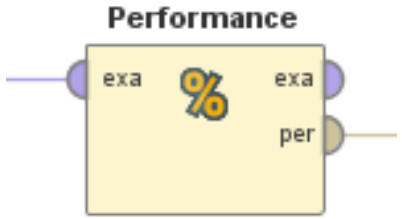
- We can also measure the cost of classification errors
  - Classifying a good part as broken costs the manufacturing cost
  - Classifying a broken part as good which causes an airplane to crash costs definitely more!

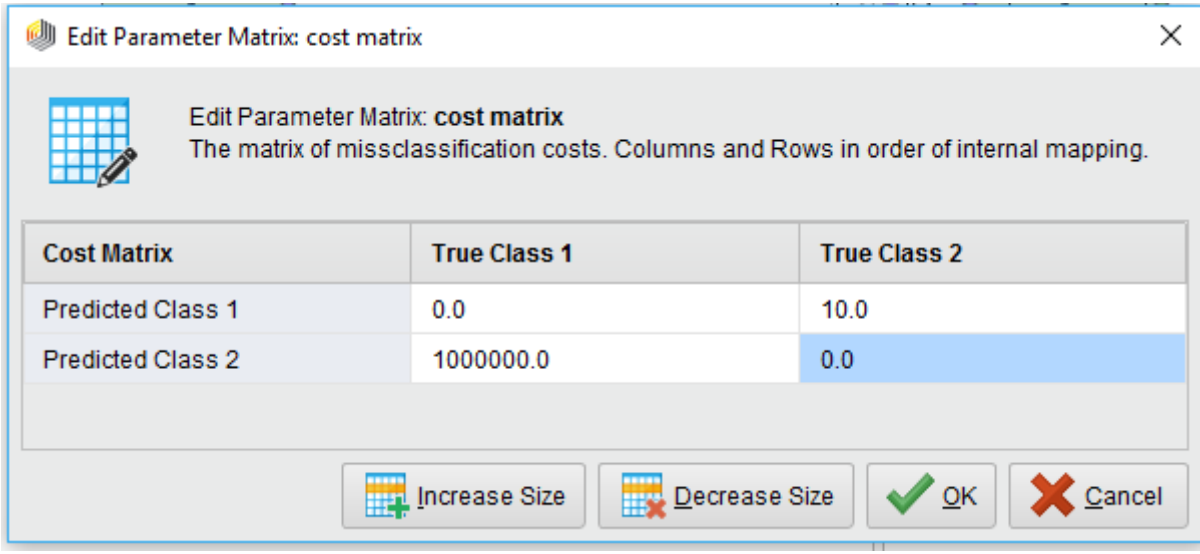
- If we sum up the cost of the errors, we get an estimation of the total cost on the test data

Operators: Performance (Costs)

- Input Port
  - Labelled Example Set
- Output Ports
  - Original Input data
  - Performance Vector
- Parameters
  - Cost Matrix
  - Class Order

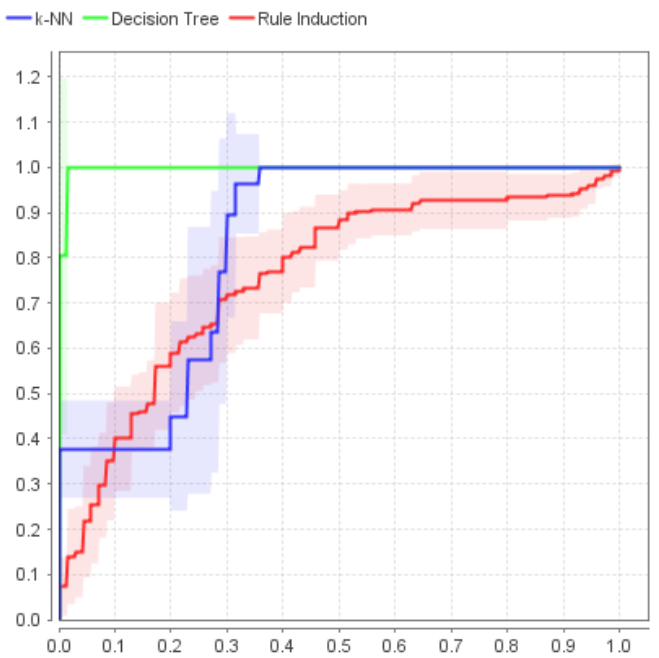






Comparing Classification Results

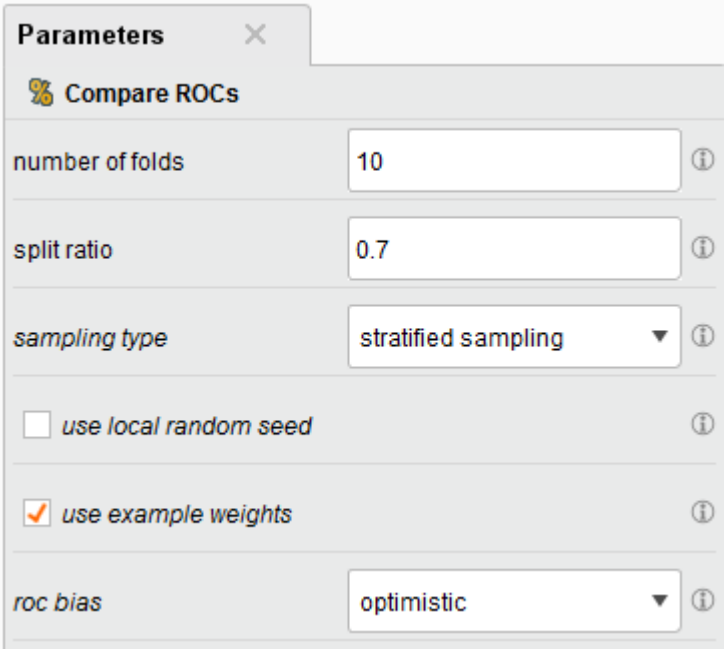
- Receiver-Operating Characteristic (ROC) curve
  - Predictions ordered by confidence
  - Correct predictions increase steepness, incorrect predictions reduce it
  - Random guessing results in diagonal
  - Interpretation: Curve A above B means algorithm A better than B
  - See also: Area Under the Curve (AUC)

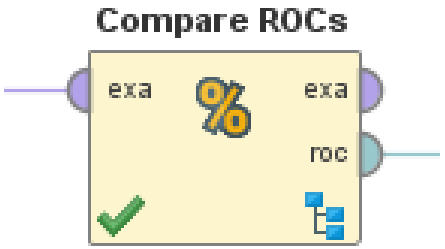


- Definitions
  - Only for binominal classification!
  - X-axis: False positive rate (=Fall-out)
    - False positive / actual negative
  - Y-axis: True positive rate (=Recall)
    - True positive / actual positive

Operators: Compare ROC

- Input Port
  - Training data (Example Set)
- Output Port
  - Training data (Example Set)
  - ROC comparison
- Parameters
  - Number of folds (Cross-Validation)
  - Split ratio (Split-Validation)
    - Used if Number of folds = -1
  - ROC bias
    - Optimistic: correct predictions first
    - Pessimistic: wrong predictions first
    - Neutral: alternate





Operators: Compare ROC (nested process)

- Add learning operators for all models that should be compared