Intro to Machine Learning

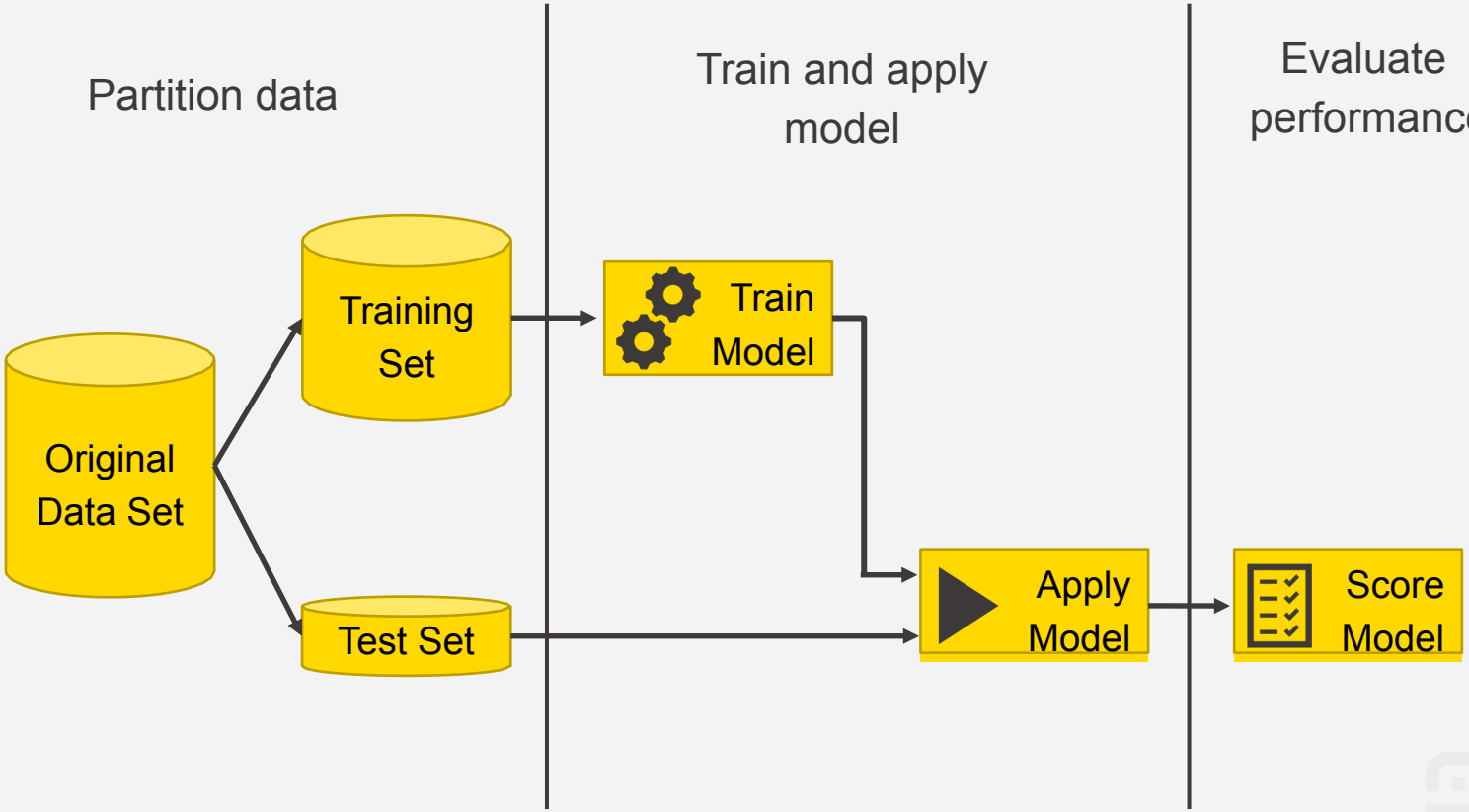
Data Mining
Partition, Learn, Predict, Score

Data Mining Strategies

Example Applications:

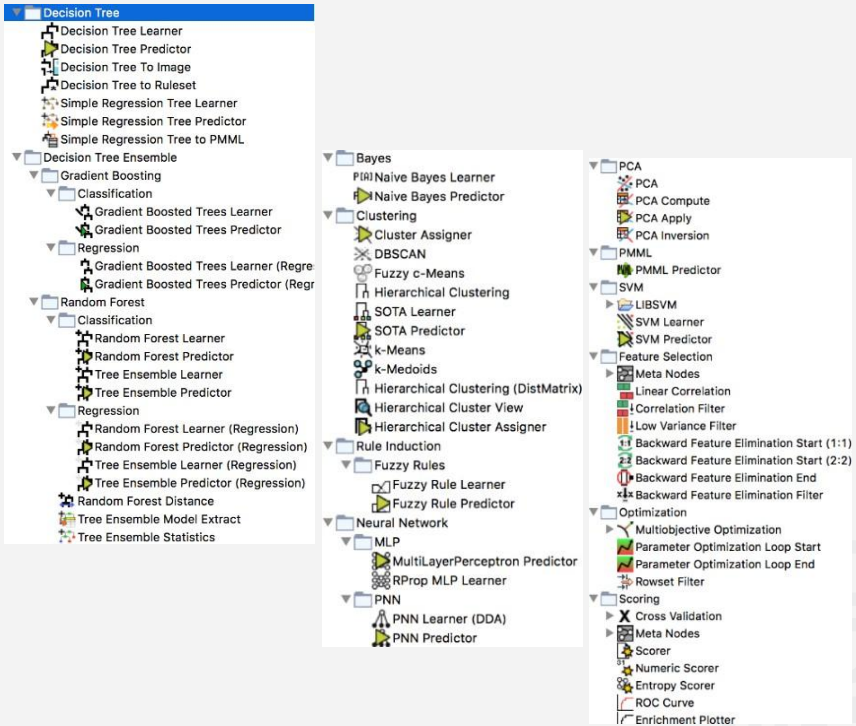
- Anomaly Detection (fraud, predictive maintenance)
- Association Rule Learning (market basket analysis)
- Clustering (market segmentation)
- Classification (next best offer, churn preventions)
- Regression (trend estimation)

Data Mining: Process Overview



Data Mining in KNIME

- KNIME has many modeling tools!
  - Decision tree, random forest, SVM, regression,
    neural networks, clustering, …
  - and integrations with other libraries:
    R, Python, H2O, WEKA, libSVM, etc.
- And many model evaluation nodes
  - ROC, standard, numeric and entropy scorers
  - Feature elimination
  - Cross validation



Churn Prediction

CRM System

Data about your customer

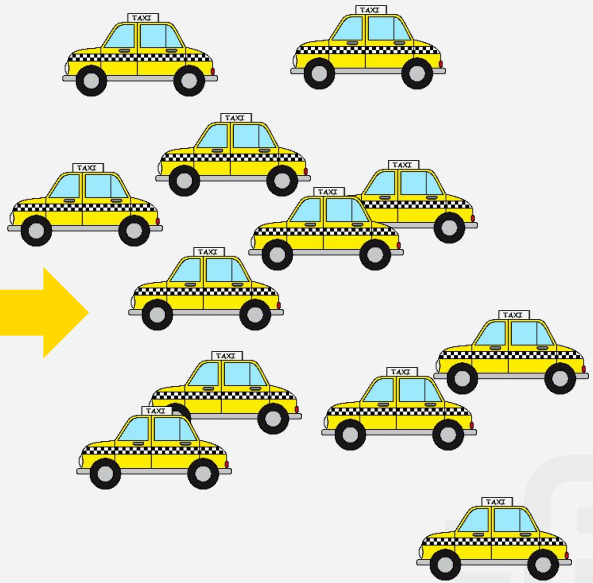
- Demographics
- Behavior
- Revenues

Model

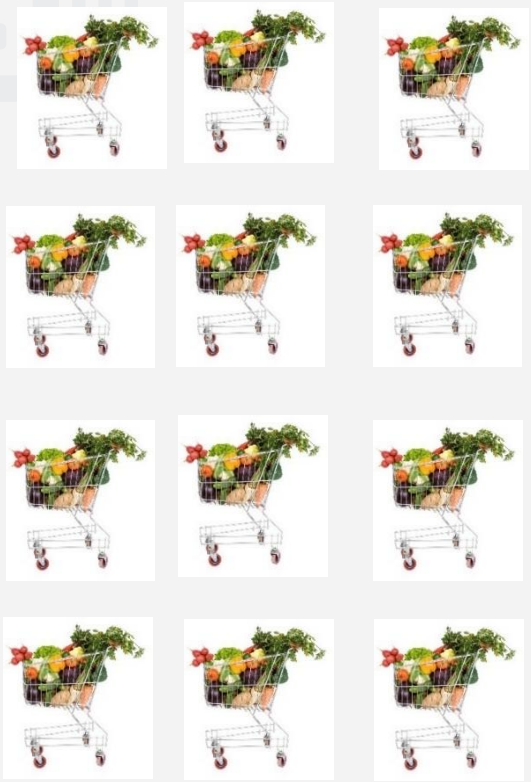
- Churn Prediction
- Upselling Likelihood
- Product Propensity /NBO
- Campaign Management
- Customer Segmentation
- …

Demand Prediction

How many taxis do I need in NYC on Wednesday at 12:00?



Recommendation Engines / Market Basket Analysis



Model

$$Rule:\ X \Rightarrow Y$$

$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

Recommendation

IF  => 

Sentiment Analysis

Samsung

Samsung Galaxy S7 Edge G935A 32GB Unlocked - Gold Platinum

4.4 out of 5 stars 125 customer reviews | 606 answered questions

5.0 out of 5 stars **Beautiful phone from a wonderful seller!**

By A May 29, 2017

Color: Gold | **Verified Purchase**

This practically new beautiful phone well exceeded my expectations!

1.0 out of 5 stars **One Star**

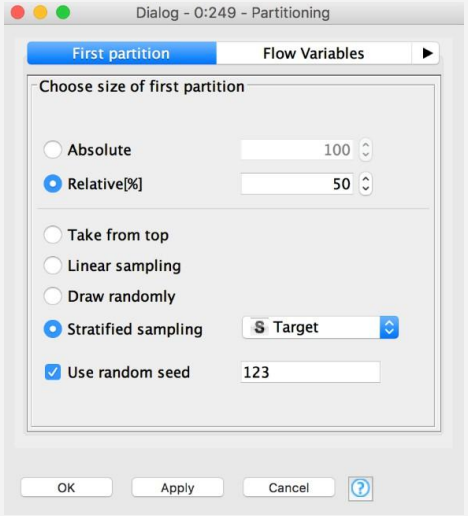
By (on August 3, 2016

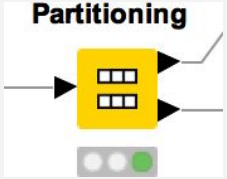
Color: Black Onyx | **Verified Purchase**

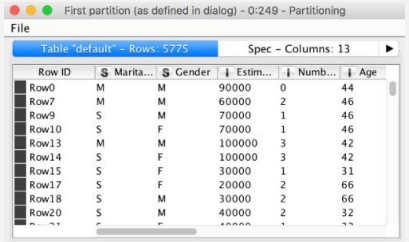
Very bad experience

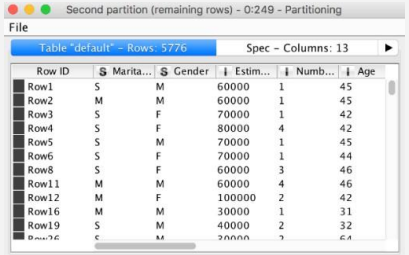
New Node: Partitioning

- Use to split data into training and evaluation sets
  - Partition by count (e.g. 10 rows) or fraction (e.g. 10%)
  - Sample by a variety of methods; random, linear, stratified



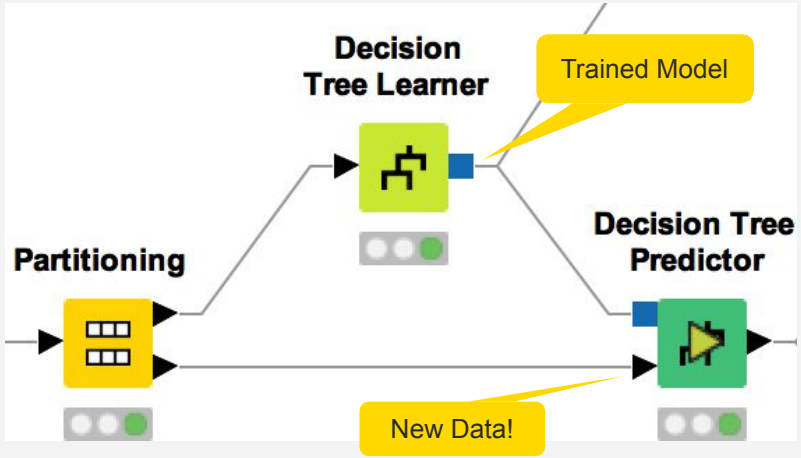






Learner-Predictor Motif

- Most data mining approaches in KNIME use a Learner-predictor motif.
- The Learner node trains the model with its input data.
- The Predictor node applies the model to a different subset of data.

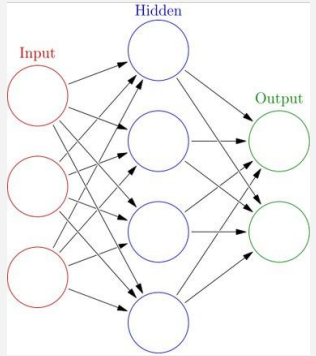


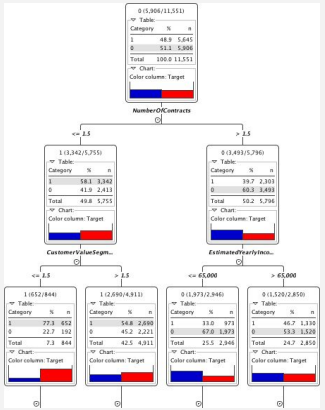
Classification

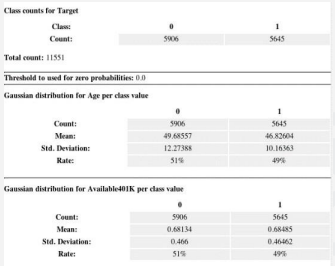
Predict *nominal* outcomes on existing data (supervised)

- Applications
  - Churn analysis (yes/no)
  - Chemical activity (active/inactive)
  - Spam detection (spam/not spam)
  - Optical character recognition (A-Z)

- Methods
  - Decision Trees
  - Neural Networks
  - Naïve Bayes
  - Logistic Regression







Class counts for Target

Class:	0	1
Count:	5906	5645

Total count: 11551

Threshold to used for zero probabilities: 0.0

Gaussian distribution for Age per class value

	0	1
Count:	5906	5645
Mean:	49.68557	46.82604
Std. Deviation:	12.27388	10.16363
Rate:	51%	49%

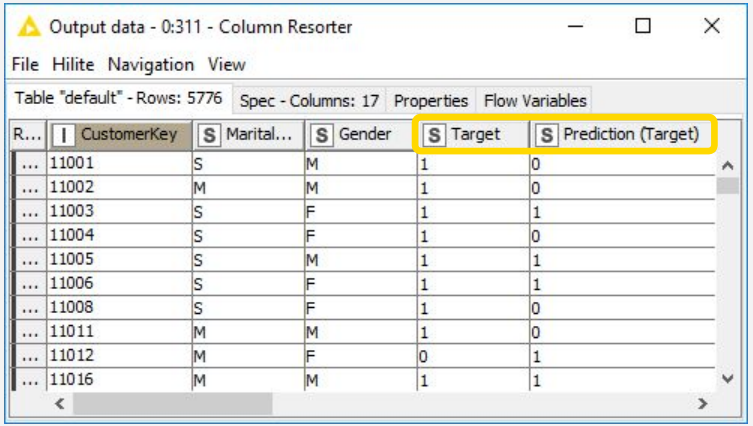
Gaussian distribution for Available401K per class value

	0	1
Count:	5906	5645
Mean:	0.68134	0.68485
Std. Deviation:	0.466	0.46462
Rate:	51%	49%

Target Column

- Target column contains values that are predicted by the classification model
- Binomial target values are often encoded to 1 and 0

Application	Target Column	Target Values
Churn analysis	Churn	Yes/No or 1/0
Chemical activity	Active	Yes/No or 1/0
Spam Detection	Spam	Yes/No or 1/0
Optical Character Recognition	Character	A-Z



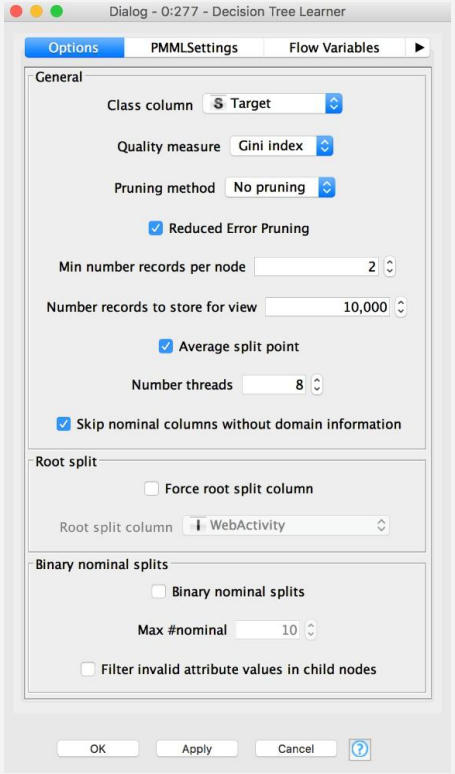
KNIME's Decision Tree

J.R. Quinlan, "C4.5 Programs for machine learning"

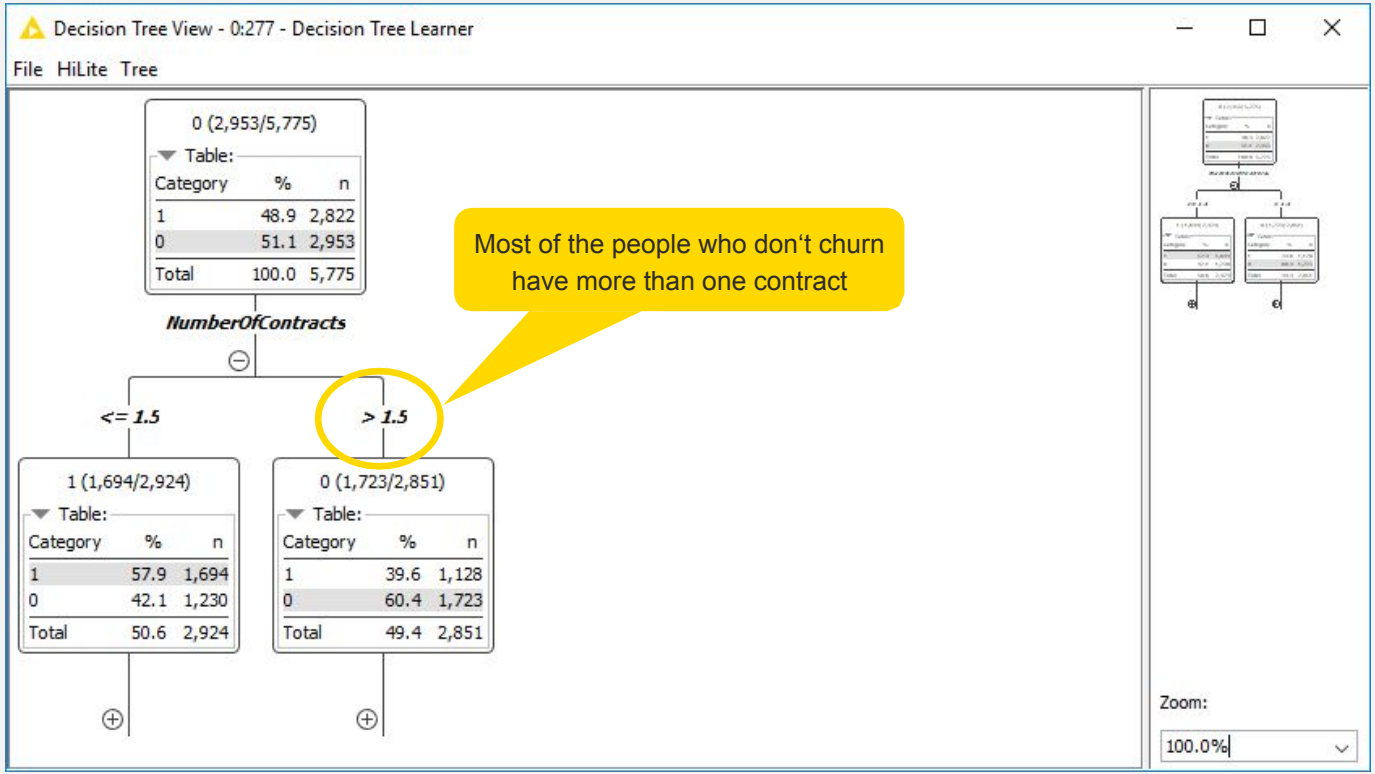
J. Shafer, R. Agrawal, M. Mehta, "SPRINT: A Scalable Parallel Classifier for Data Mining"

- C4.5 builds a tree from a set of training data using the concept of information entropy.
- At each node of the tree, the attribute of the data with the highest **normalized information gain** (difference in entropy) is chosen to split the data.
- The C4.5 algorithm then recurses on the smaller sub lists.

New Node: Decision Tree Learner

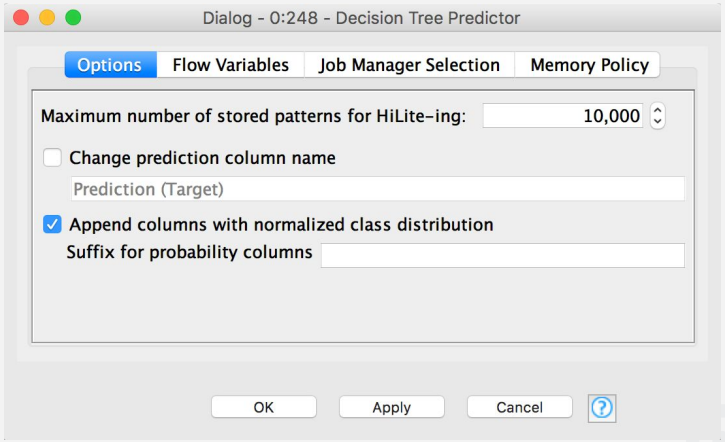


Decision Tree View



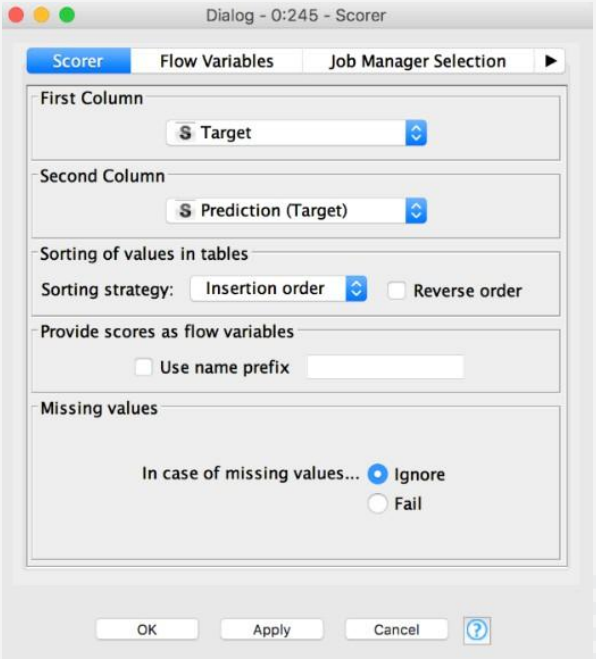
New Node: Decision Tree Predictor

- Takes a decision tree model & applies it to new data
- Check the box to append class probabilities



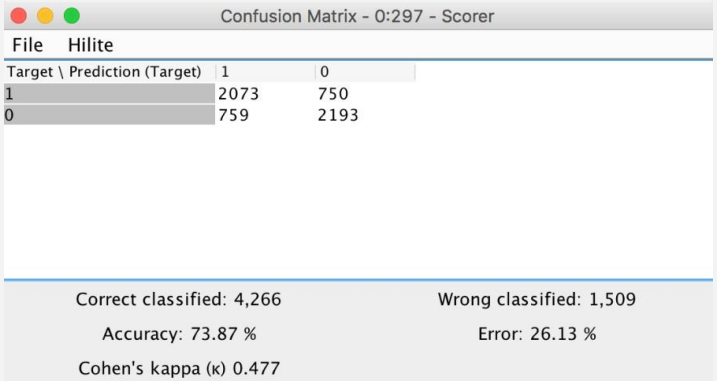
New Node: Scorer

Compare predicted results to known truth
in order to evaluate model quality

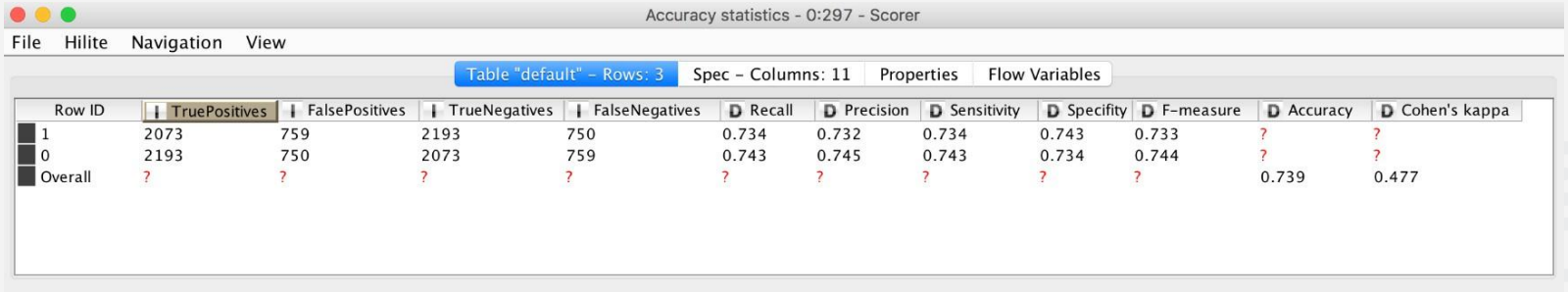


New Node: Scorer

- Confusion matrix shows the distribution of model errors

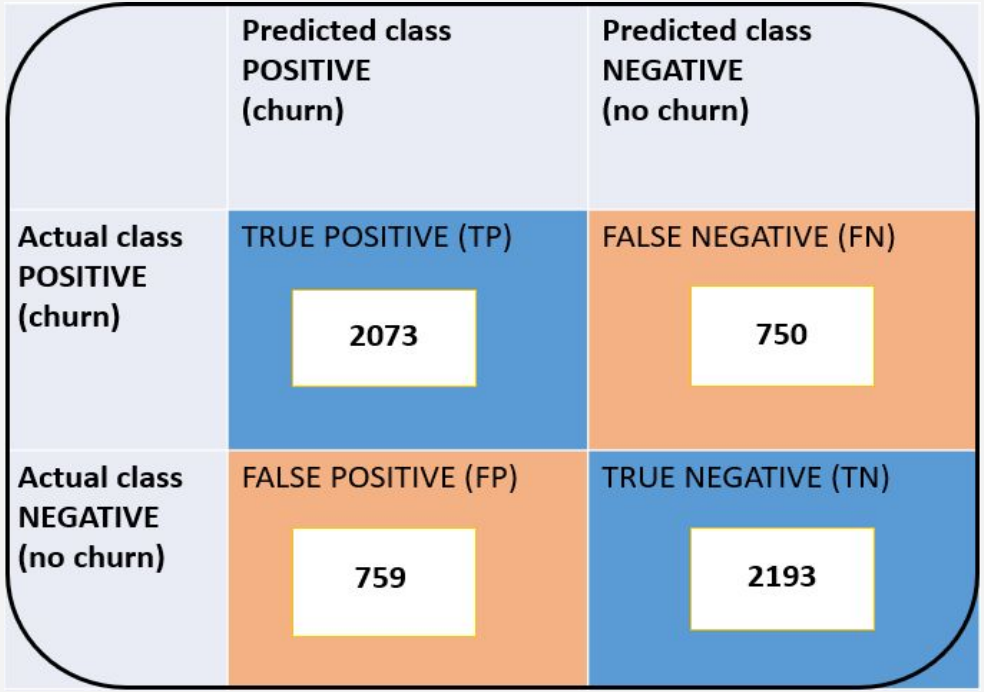


- An accuracy statistics table provides a detailed analysis of model quality



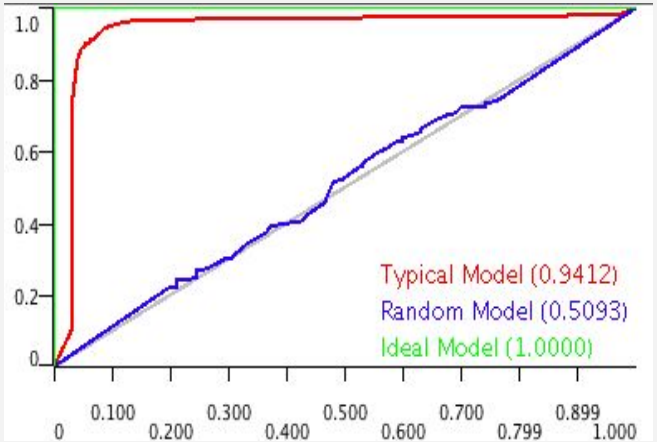
Row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specificity	F-measure	Accuracy	Cohen's kappa
1	2073	759	2193	750	0.734	0.732	0.734	0.743	0.733	?	?
0	2193	750	2073	759	0.743	0.745	0.743	0.734	0.744	?	?
Overall	?	?	?	?	?	?	?	?	?	0.739	0.477

Confusion Matrix



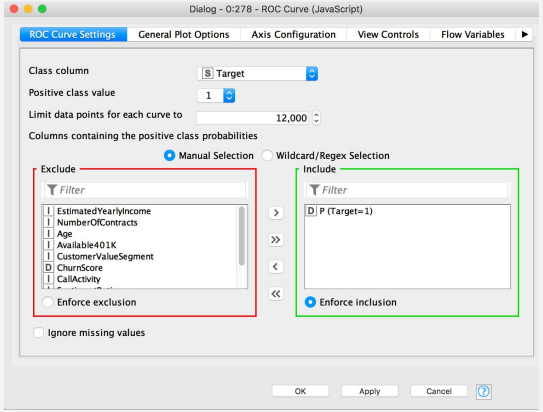
Receiver Operating Characteristics

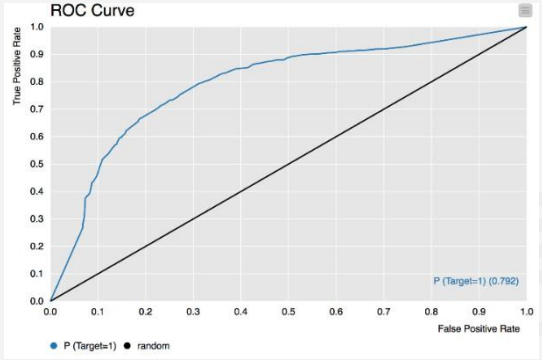
- Sort by confidence in target class
- Plot true positive rate vs false positive rate
- Ideal models achieve 100% TPR with 0% FPR
- Area under the curve indicates model quality
  - (1=ideal model, 0.5 = random outcome)

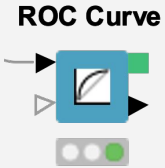


New Node: ROC Curve

- Requires individual class probabilities from a preceding predictor
- User must define:
  - Original class column
  - Positive class value
  - Probability for the selected positive class value for one or multiple models







Data Mining Exercise, Activity I

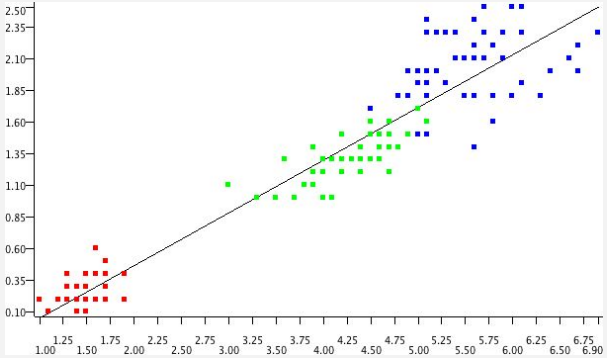
Start with exercise: *Data Mining, Activity I*:

- Partition the fully joined data into a training and test set (50%, Stratified Sampling on Target)
- Train a decision tree on the training set to predict Target
- Use the trained model to predict Target in the test set
- What is the overall accuracy of your model?
- Optional: Evaluate the accuracy and robustness of the model with the ROC Curve node

Regression

Predict *numeric* outcomes on existing data (supervised)

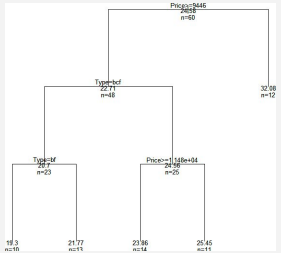
- Applications
  - Forecasting
  - Quantitative Analysis
- Methods
  - Linear
  - Polynomial
  - Regression Trees
  - Partial Least Squares



Statistics on Linear Regression

Variable	Coeff.	Std. Err.	t-value	P>|t|
Petal.Length	0.4158	0.0096	43.3872	0.0
Intercept	-0.3631	0.0398	-9.1312	4.44E-16

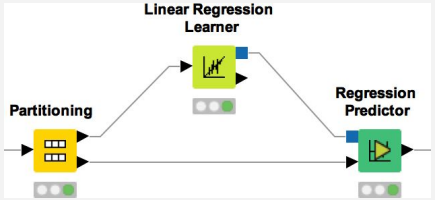
Multiple R-Squared: 0.9271
Adjusted R-Squared: 0.9266

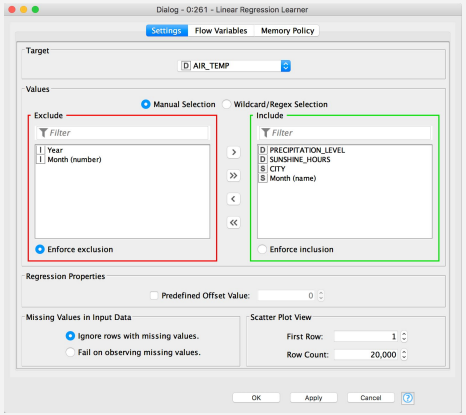


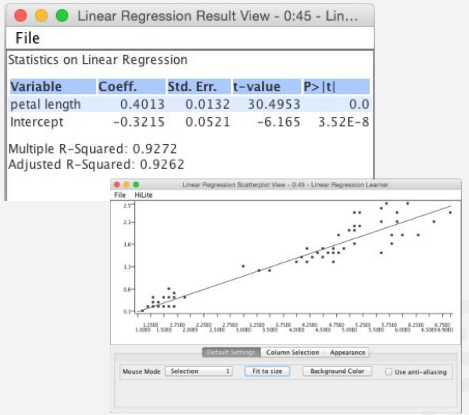
New Nodes: Linear Regression Learner & Regression Predictor

A linear model relating a dependent variable to 1 or more independent variables

- Model coefficients provided in 2nd output port
- Also available: Polynomial and Tree Ensemble Regression nodes



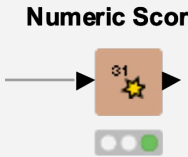


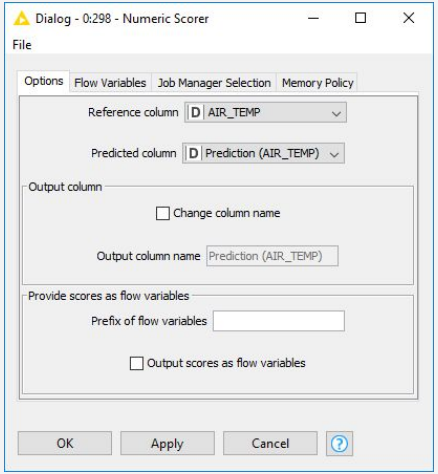


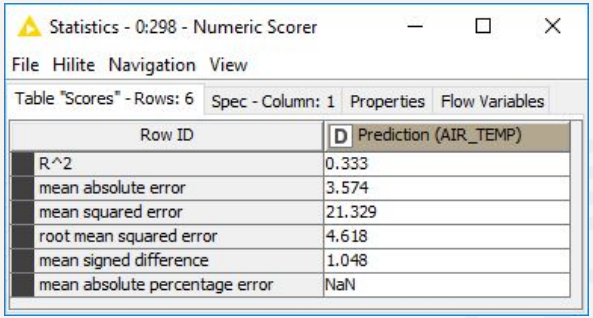
New Node: Numeric Scorer

Similar to scorer node, but for nodes with *numeric* predictions (e.g. linear/polynomial regression)

- Compare dependent variable values to predicted values to evaluate goodness of fit.
- Report R^2, MAE, MSE, RMSE etc.





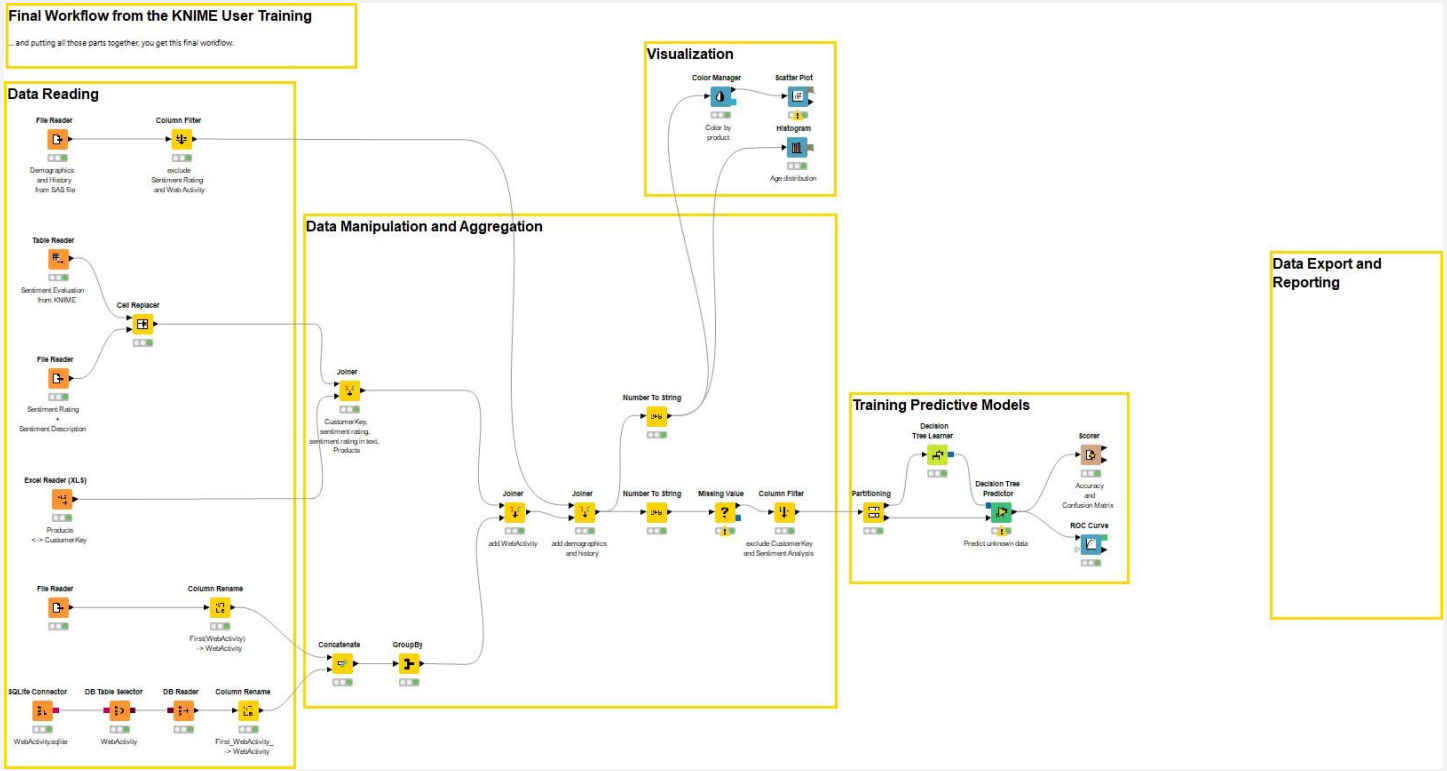


Row ID	Prediction (AIR_TEMP)
R^2	0.333
mean absolute error	3.574
mean squared error	21.329
root mean squared error	4.618
mean signed difference	1.048
mean absolute percentage error	NaN

Data Mining Exercise, Activity II

- Start with exercise: *Data Mining, Activity II*:
- Read *weather.table* data
- Split the data into rows up to 2016 (training set) and rows from 2017 on (test set)
- Train a linear regression model that predicts the AIR_TEMP as a function of all other features in the dataset
- Use the model to predict the temperature in 2017 and evaluate the model with the Numeric Scorer node
- Optional:
  - Calculate the mean temperature per month in the training data
  - Join the mean temperature per month to the test set
  - Use the Numeric Scorer to see if the average monthly temperature provides a better prediction than the Linear Regression model

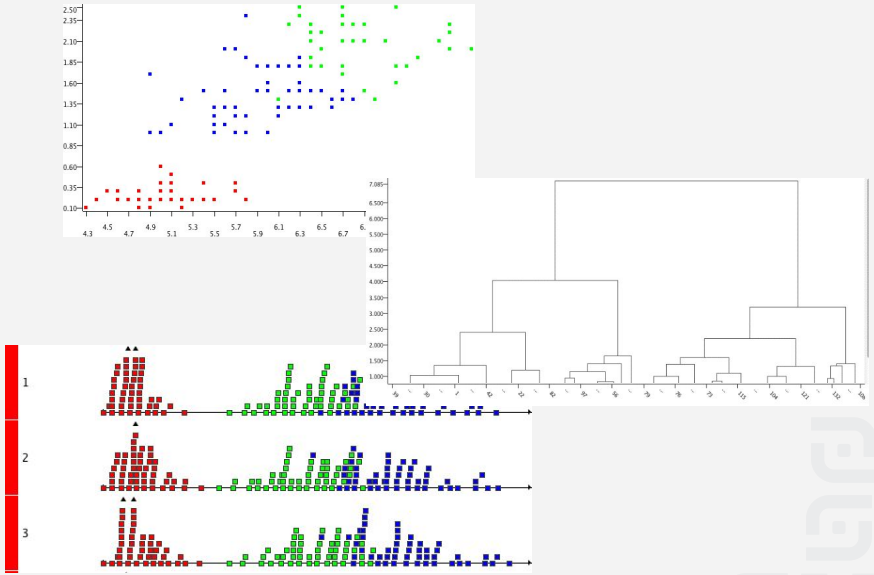
Today's Example



Clustering

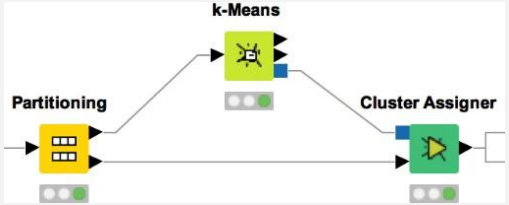
Discover hidden structure in unlabeled data (unsupervised)

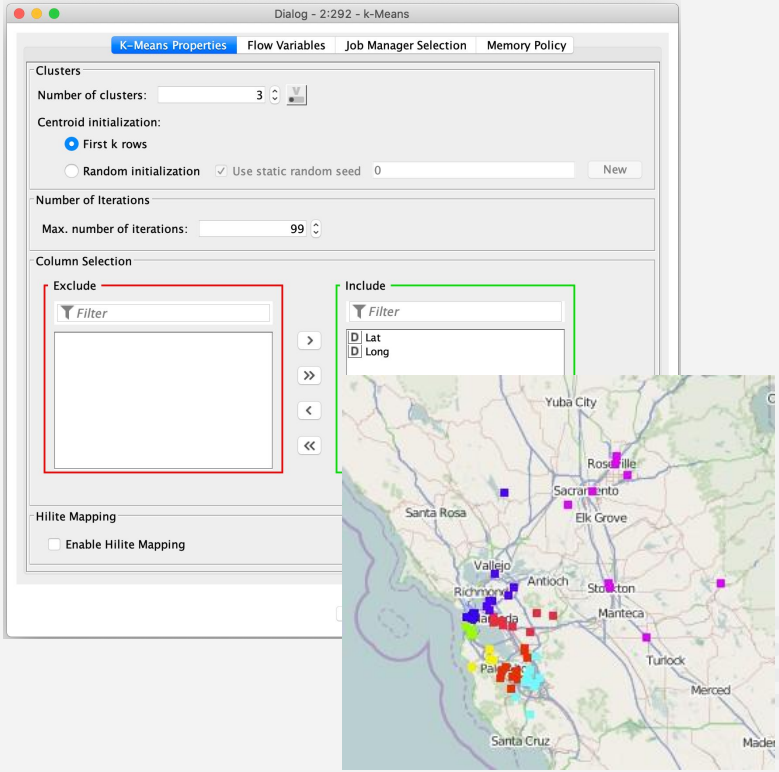
- Applications
  - Market Segmentation
  - Diversity picking
- Methods
  - K-means/medoids
  - Hierarchical
  - DBScan
  - OPTICS
  - Neighbourgrams



New Nodes: k-Means Clustering

- Looks at n observations to define the means for k clusters.
- Each observation is then assigned to its closest cluster center.
- You must provide k.





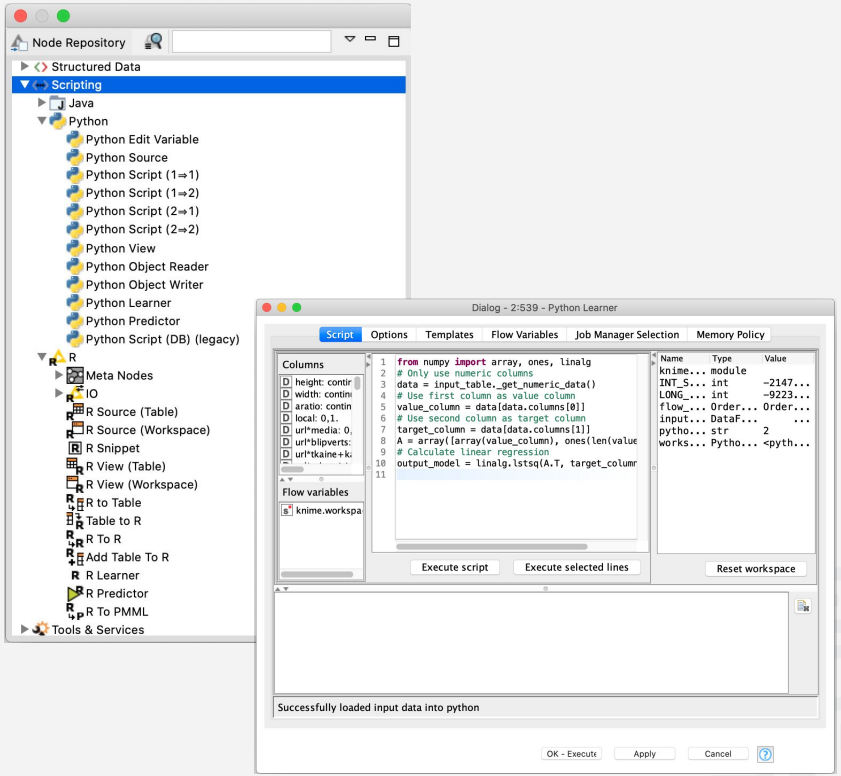
Data Mining Exercise, Activity III

Start with exercise: *Data Mining, Activity III*

- Read *location_data.table* data
- Filter the data to entries from California (region_code = CA)
- Perform k-means clustering with k=3. Use only latitude and longitude for clustering.
- Optional: plot latitude and longitude in a view (OSM Map or Scatter Plot) and use the view to visually optimize k

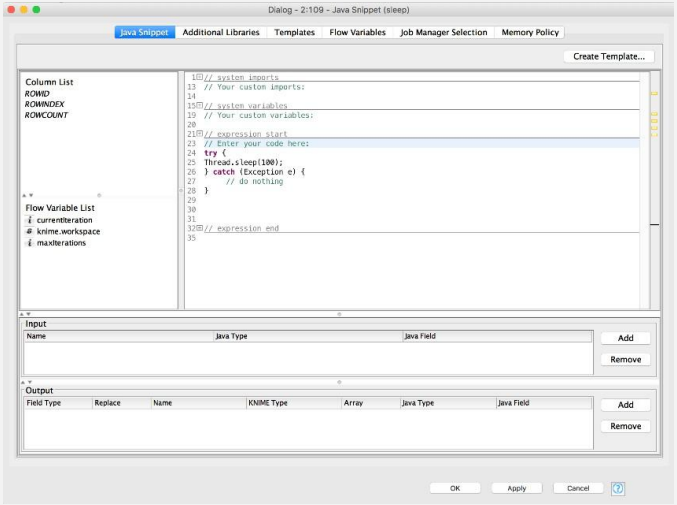
Scripting Integrations: R and Python

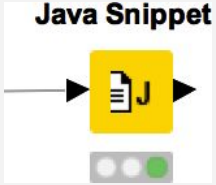
- Run R or Python code in KNIME Analytics Platform
- Works with existing Python and R installations
- Syntax highlighting support
- Different nodes for many tasks, e.g training a model using an algorithm available in Python



Java Snippet

- Fastest running scripting node in KNIME
- Syntax highlighting, auto completion, error checking
- Templates allow you to save scripts for later re-use
- Import custom libraries





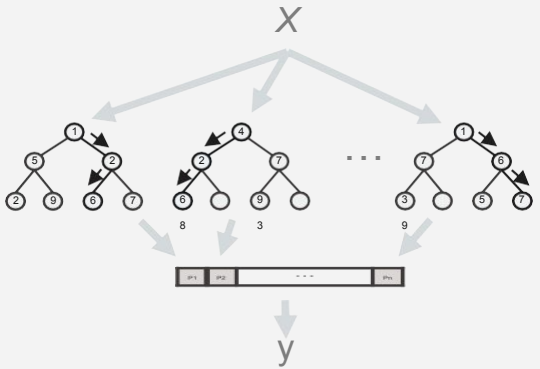
Advanced Data Mining

Overview

- Ensemble models
  - Random Forest / Tree Ensembles
  - Gradient Boosted Trees
- Parameter optimization
- Cross validation
- H2O and Keras integration in KNIME

KNIME's Tree Ensemble Models

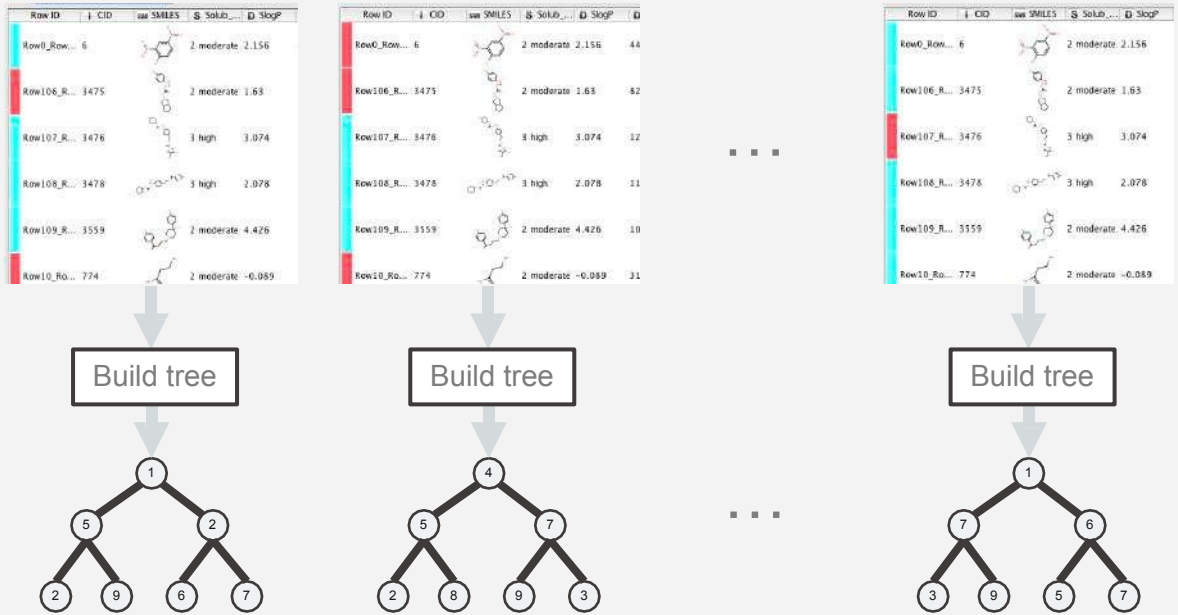
- The general idea is to take advantage of the "wisdom of the crowd"
- Ensemble models: Combining predictions from a large number of weak predictors, e.g. decision trees
- Leads to a more accurate and robust model
- This is called "bagging"



Typically: for classification the individual models vote and the majority wins; for regression, the individual predictions are averaged

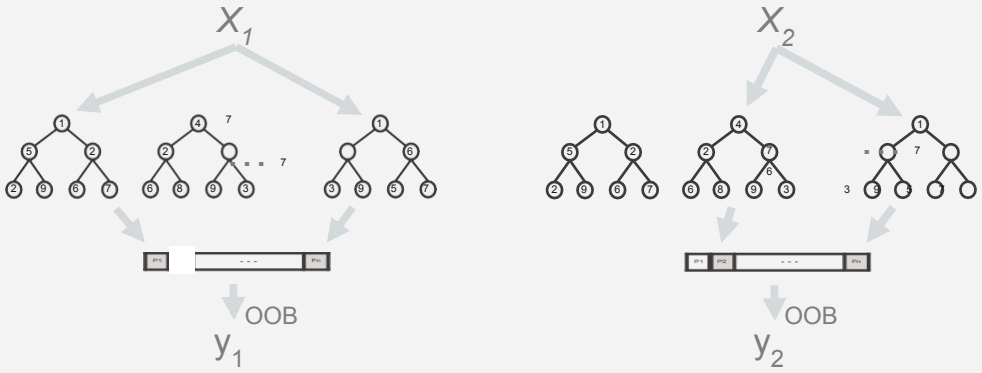
How Does Bagging Work?

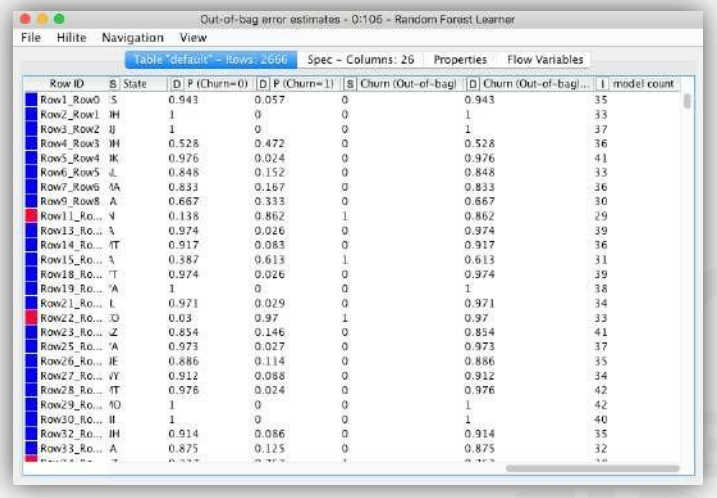
- Pick a different random subset of the training data for each model in the ensemble (bag)



An Extra Benefit of Bagging: Out of Bag Estimation

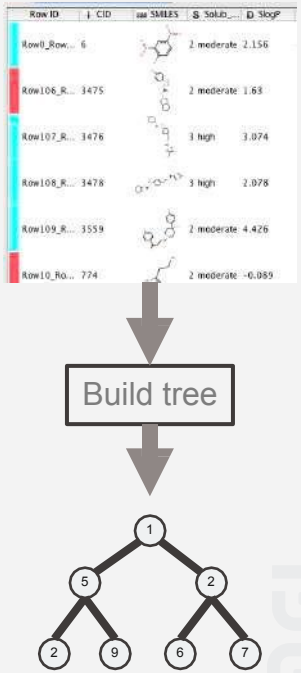
- Allows testing the model using the training data: when validating, each model should only vote on data points that were not used to train it





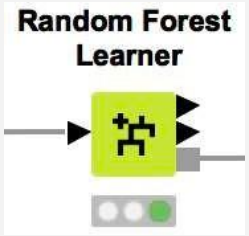
Random Forest

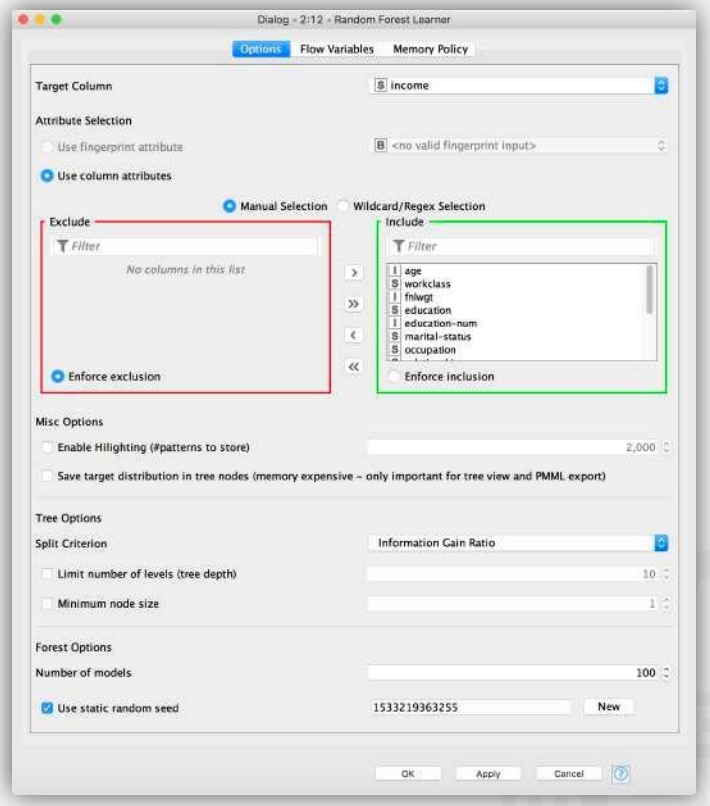
- Train a bag of decision trees
- For each tree / model a training set is generated by sampling uniformly with replacement from the standard training set
- An extra element of randomization is used when building the trees : **each node** in the decision tree only “sees” **a subset of the input columns**, typically $\sqrt{N}$
- Random forests tend to be very robust w.r.t. overfitting (though the individual trees are almost certainly overfit)
- Extra benefit: training tends to be much faster



Random Forest Learner

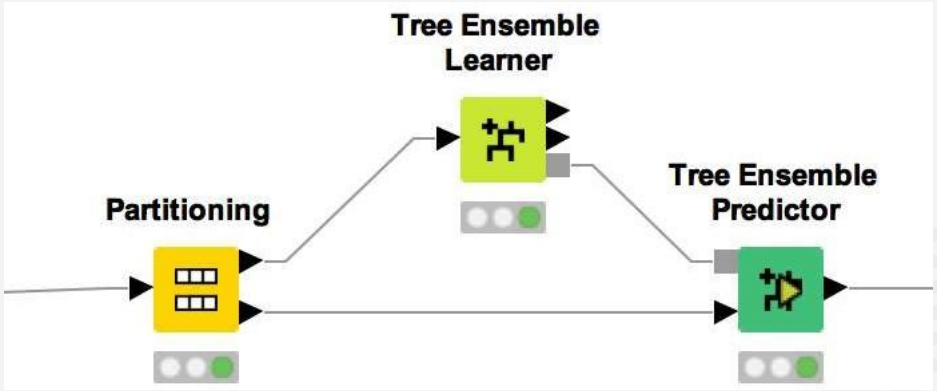
- The output model describes a random forest and is applied in the corresponding predictor node using a simple majority vote
- The statistics table on the attributes tells how often each attribute…
  - … is used in the first three splits
  - … was a possible candidate in the first three splits





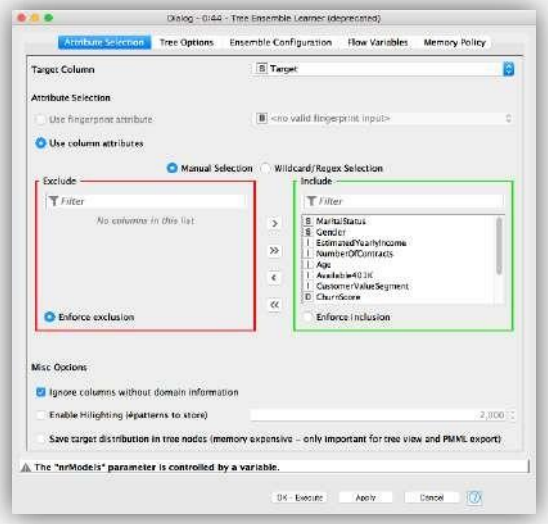
Tree Ensembles

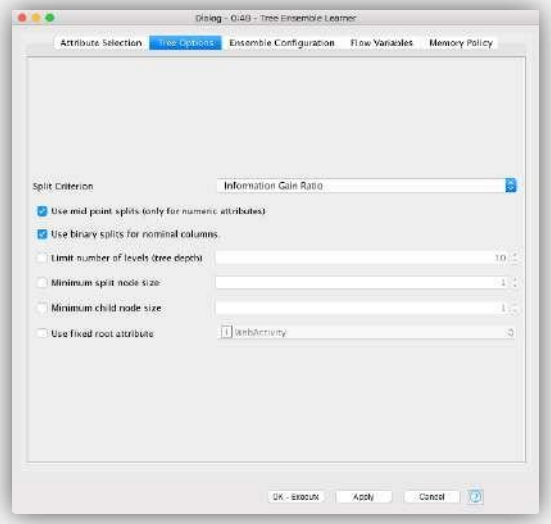
- Random Forest is a specific tree ensemble with predefined ensemble parameters
- The Tree Ensemble Learner node allows to train different tree ensembles
  - E.g. different row and column sampling options
- Optimization of a tree ensemble is complex due to a surplus of configuration options
  - Number of models
  - Number of columns
  - Number of rows
  - Tree depth
  - ...

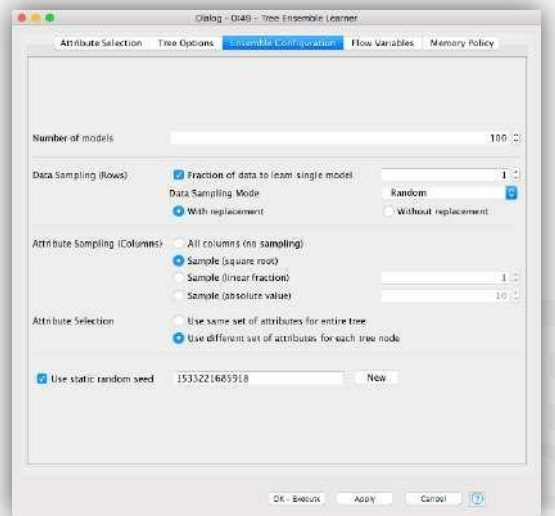


Tree Ensemble Learner

- Choose which columns to include
- Configure a prototype tree (depth, split criteria etc.)
- Setup ensemble parameters (model count, row/column subsampling)

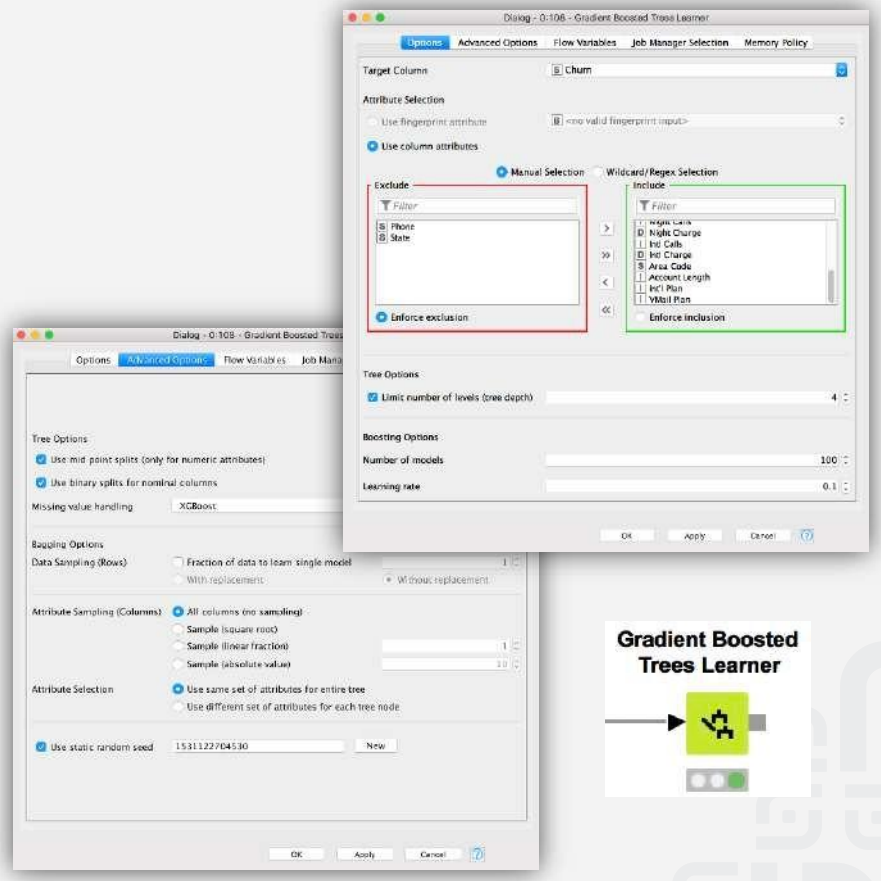






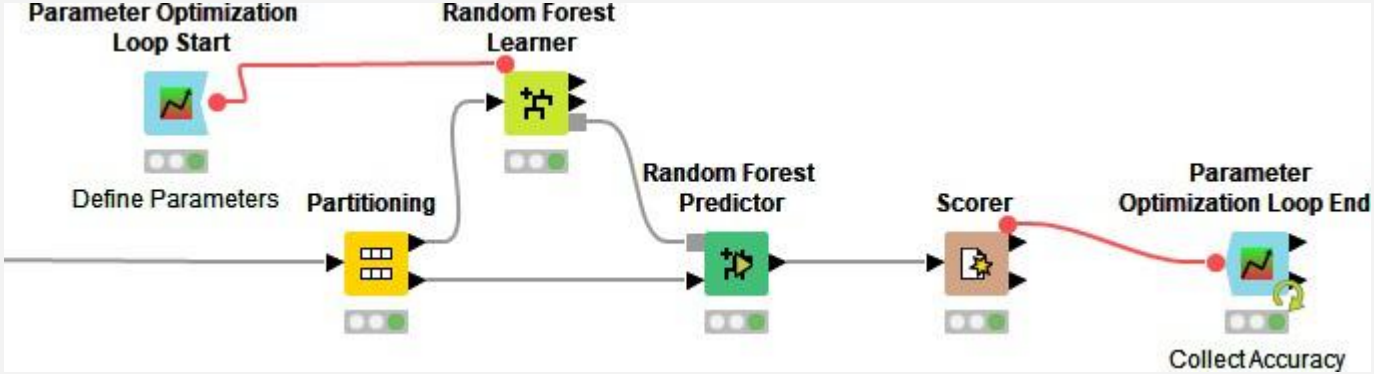
Gradient Boosted Trees Learner

- Another algorithm for creating ensembles of decision trees
- Starts with a shallow tree
- Builds additional trees to fit the residual errors
- Can introduce randomness in choice of data subsets (“stochastic gradient boosting”) and in variable choice (Advanced Options)



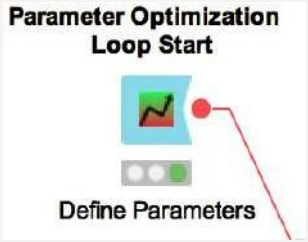
Parameter Optimization

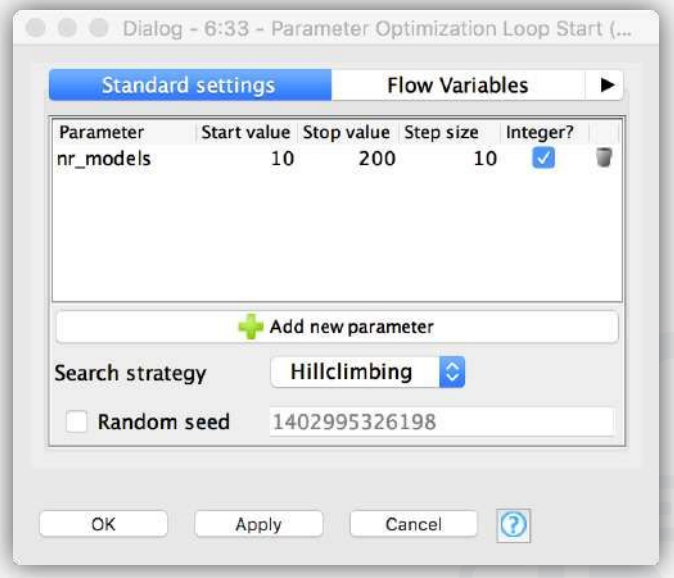
- Some modeling approaches are very sensitive to their configuration.
- Calculating optimum settings is not always possible.
- Parameter Optimization loops may help find a good configuration.



Parameter Optimization Loop Start

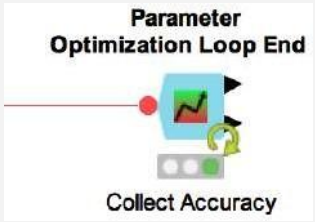
- Define parameters to optimize
- Set upper/lower bounds and step sizes (and flag integers)
- Choose an optimization method
  - Brute force for maximum accuracy, but slower computation
  - Hillclimbing for better faster runtimes, but may get stuck in local optimum settings
  - Random search to randomly search for parameter values within a given range
  - Bayesian Optimization (TPE)

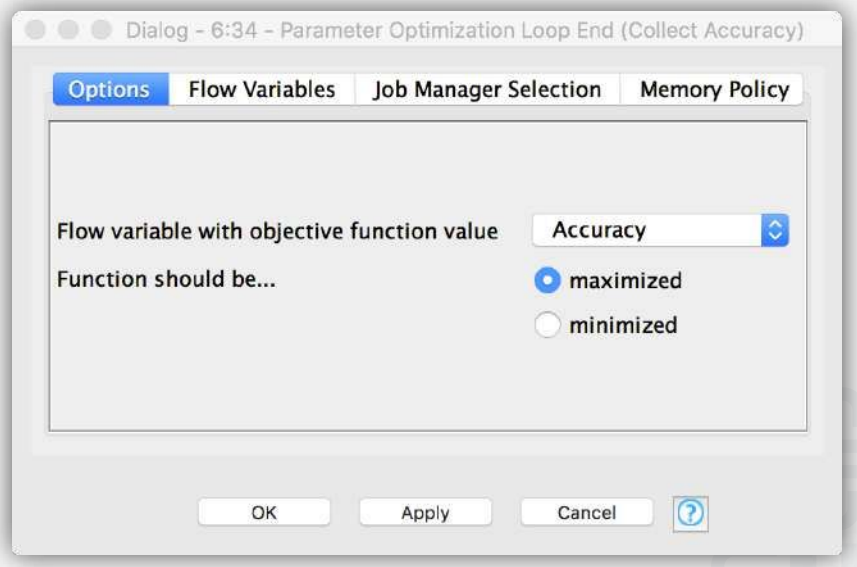




Parameter Optimization Loop End

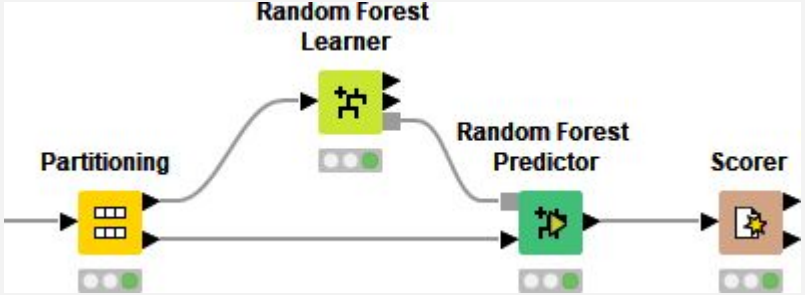
- Collects a value to optimize as Flow Variable.
- Value may be maximized (accuracy) or minimized (error)

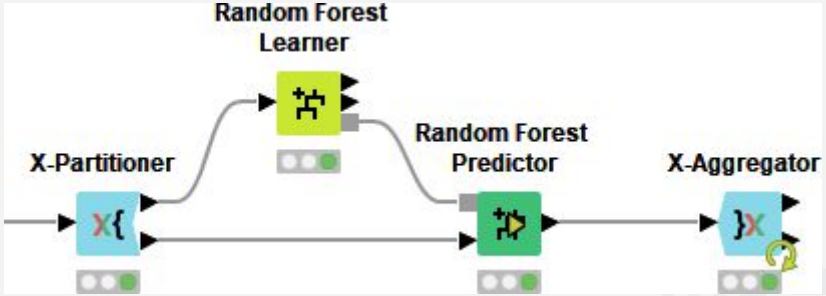




Cross Validation

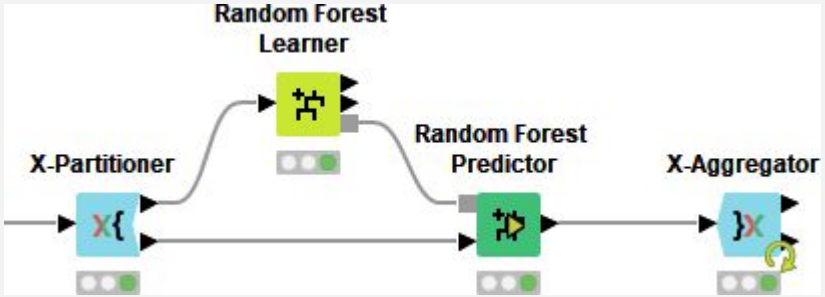
- Used to evaluate model stability
- Re-execute the modeling process many times using different data partitions
- Collect aggregated statistics on model accuracy

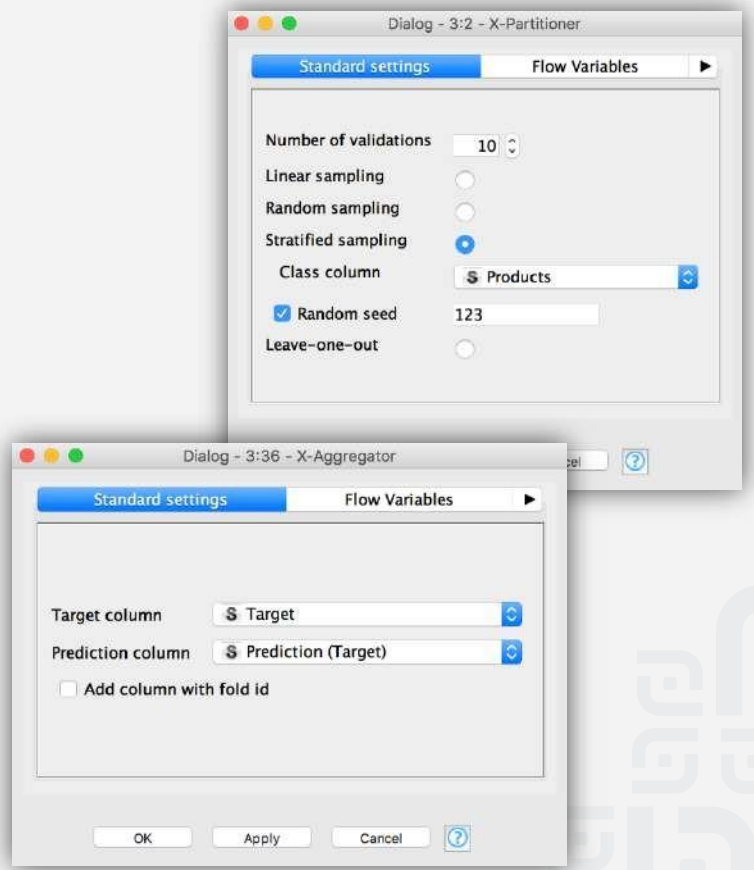




Example: Cross Validation

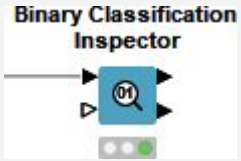
- X-Partitioner X-Aggregator
- X-Partitioner replaces Partition
- X-Aggregator replaces Scorer
- Can be used with any learner/predictor





Binary Classification Inspector

- Inspect and compare the performances of classification models
- Adjust the classification threshold according to the goal of your model



Select the scoring metrics to compare

Compare the ROC curves

Inspect the distribution of the predicted positive class probabilities

Inspect the class assignments for different classification thresholds

Advanced Data Mining Exercise

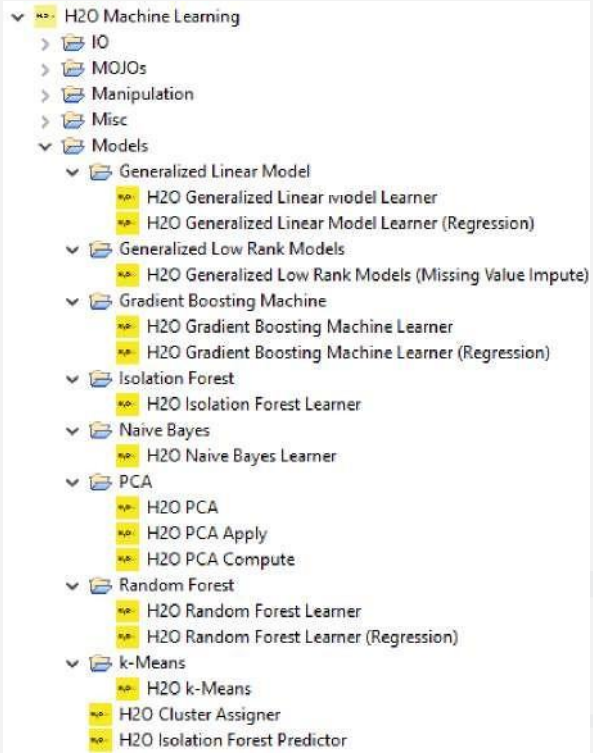
Start with exercise: *Advanced Data Mining*

Goal: Train a decision tree and a random forest model and compare their performance

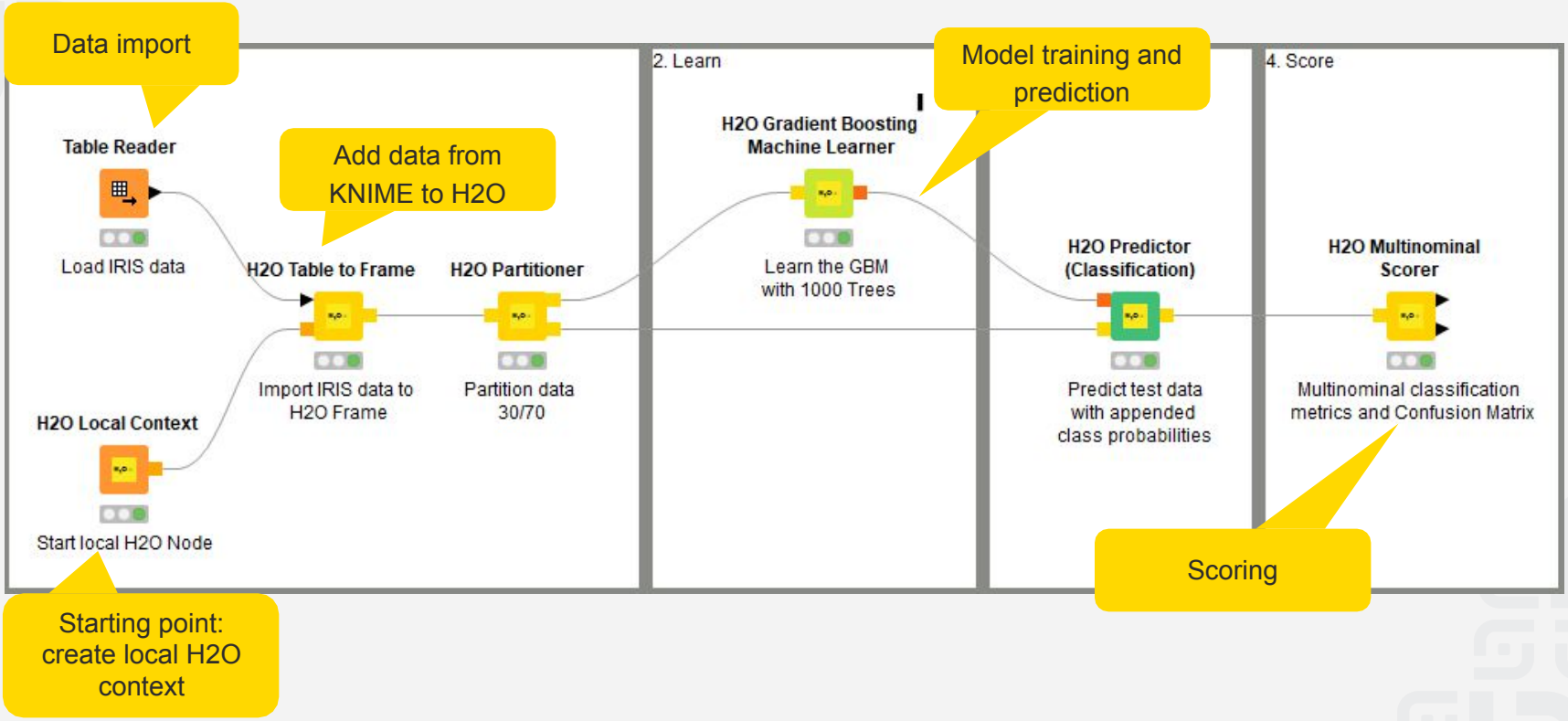
- Partition the data 50/50 using stratified sampling on the "Target" column
- Train and apply a Random Forest model to predict the "Target" column
- Train and apply a Decision Tree model to predict the "Target" column
- Combine the performances of both models (Column Appender node)
- Evaluate the performances of the models (Binary Classification Inspector node)
  Which model performs better?

H2O Integration

- KNIME integrates the H2O machine learning library
- H2O: Open source, focus on scalability and performance
- Supports many different models
  - Generalized Linear Model
  - Gradient Boosting Machine
  - Random Forest
  - k-Means, PCA, Naive Bayes, Isolation Forest, etc. and more to come!
- Includes support for MOJO model objects for deployment

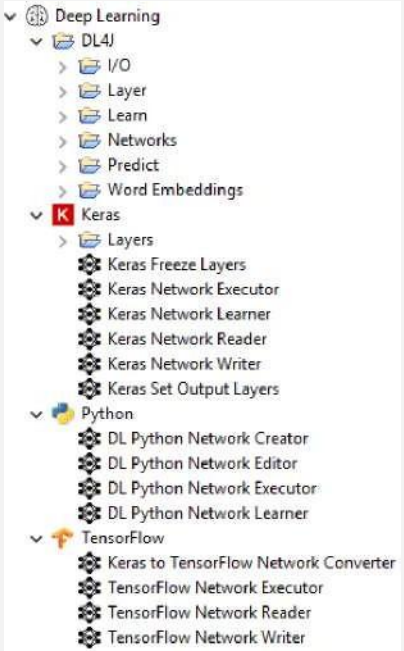


H2O Integration - Example

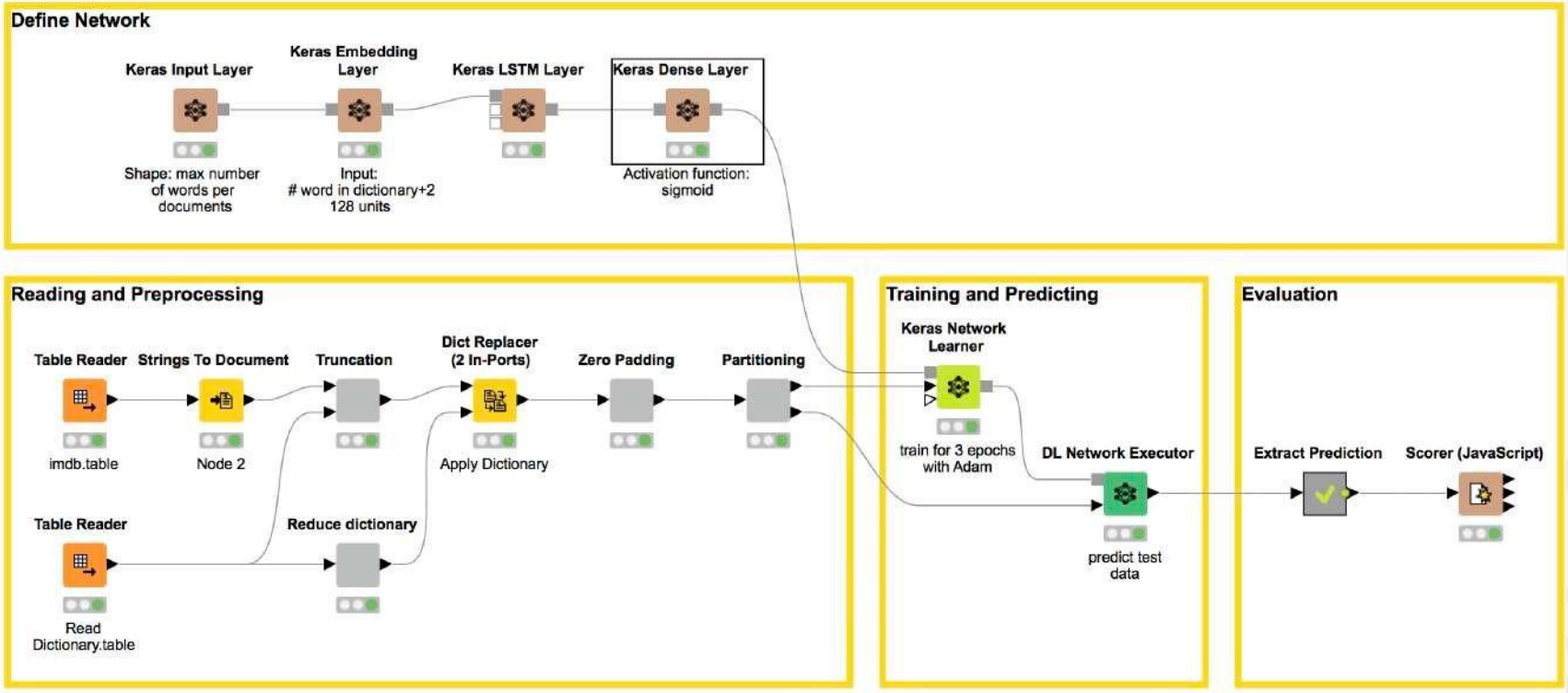


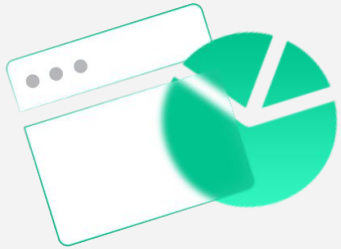
Deep Learning Integration

- Keras integration:
  - Many different layer nodes.
  - Define your network, train and apply a network without a single line of code.
- DL Python integration
- TensorFlow integration



Sentiment Analysis Using Keras





Terima Kasih