izMiR: computational ab initio microRNA detection

Jens Allmer (jens@allmer.de)
jLab
Müşerref Duygu Saçar Demirci
jLab

Method Article

Keywords: MicroRNA detection, computational, ab initio, machine learning

Posted Date: July 15th, 2016

DOI: https://doi.org/10.1038/protex.2016.047

Abstract

izMiR is an data analysis workflow that can be used for the detection of pre-miRNAs. The overall system includes two important workflows 1) Training of machine learning classifiers with suitable examples 2) Application of the learned model for detection of new pre-miRNAs. By using the training workflow it is possible to generate models that can be used on new data to predict whether the given data have potential miRNA hairpins. It is also possible to use prediction workflow directly with the models and input data provided by us. In the latter case compatible features need to be calculated for analysis. These calculations can be done using two webservers. One provided by us \(http://www.jlab.iyte.edu.tr/software/izmir) and one by Yones et al. \(http://www.fich.unl.edu.ar/sinc/web-demo/mirnafe-full/).

Introduction

In many machine learning based miRNA precursor prediction studies, different data sets were used with various features using different machine learning algorithms. Superficially comparing the published performance measures is at best misleading for the end users. To allow for a proper comparison, these tools need to be unified in one framework and tested on the exact same inputs. There are many challenges for such ab initio methods. Here we provide a comprehensive approach which allows the opportunity to compare different data sets, feature groups and classifiers. The system is very flexible and can be seamlessly adopted for future studies gracefully allowing extensions and adjustment of any settings. We also showed that by using izMiR it is possible to obtain consensus models which lead to increased classification performance. For both learning and prediction features need to be calculated. For this two services are available one provided by Yones et al. \(http://fich.unl.edu.ar/sinc/web-demo/mirnafe-full/) and one provided by us \(http://jlab.iyte.edu.tr/software/izmir). The main parts of learning and prediction workflows are: Training: The below steps are laid out in the KNIME workflow \(http://bioinformatics.iyte.edu.tr/software/izmir, training workflow) and the sections are labeled accordingly \(Figures 1-6). - Selection of suitable training examples - Feature calculation - Input data for positive \(real miRNA) and negative \(non-miRNA) examples \(Fig. 1) - Preprocessing of input data \(Fig. 1) - MCCV \(Monte Carlo Cross Validation; Figs. 2,3) - Sampling \(Fig. 3) - Feature grouping \(Fig. 4) - Classifier training \(Naive Bayes \(NB), Decision Tree \(DT), Weka LibSVM) per study \(Fig. 5) - Model performance scores \(e.g.: accuracy, F-score, Fig. 5) - Model generation \(Fig. 5) - Combination of outcomes for each iteration \(Fig. 4) - Model evaluation \(Fig. 5) - Model output \(PMML file) to be used in prediction phase \(Figs. 1,6) Prediction: The below steps are laid out in the KNIME workflow \(http://bioinformatics.iyte.edu.tr/software/izmir, prediction workflow) and the sections are labeled accordingly \(Figures 7-12). - Feature calculation - Input data reading and preprocessing \(Fig. 7) - Applying best PMML models \(created in training phase; Fig. 8,9) - Obtaining predictions \(miRNA or negative) and prediction score \(between 0-1, Fig. Figs 9-11) - For Consensus Decision Tree and Naive Bayes; if a sequence is predicted as “miRNA” by 6 or more studies, it is consensus result is miRNA otherwise negative \(Fig. 9) - For Consensus Rule; ; if average DT or average NB is larger than 0.89 then it

is labeled as “miRNA”, conversely, if average DT or average NB is less than 0.5 it is labeled as “negative”, finally the remaining are labeled as “candidate” miRNA. \(Fig. 10) - For Consensus Model generation the learning data sets used in this study are tested on the best models and prediction scores are saved. Then these prediction scores are used as input data for a new learning process by using Multi Layer Perceptron classifier \(1000 fold Monte Carlo Cross Validation) and the model with the highest accuracy and F-score is stored. This Consensus Model is then included in prediction workflow \(Fig. 11) - Visualization meta-node is a collection of visual nodes available in KNIME for producing simple graphs \(Fig. 12) - Count meta-node provides numeric information about the results \(Fig. 1)

Reagents

-

Equipment

1) KNIME \(https://www.knime.org) with all free extensions \(at least the Weka plugin must be installed).

Procedure

Training 1) KNIME installation a. Download: https://www.knime.org/downloads/overview b. Installation: https://tech.knime.org/installation-0 c. Update extensions: https://www.knime.org/downloads/update 2) Importing workflows - Download workflows from http://bioinformatics.iyte.edu.tr/supplements/izmir - Open KNIME - On the left side of KNIME window, there is a box with LOCAL \(Local Workspace) \(Figure 1), right click to that area and “select import KNIME workflow” \(https://tech.knime.org/workbench) - In the pop-up window select the directory where the downloaded workflows are and load them - Open the loaded workflow - The workflow has the input data \(human miRNAs as positive and pseudo as negative) If you do not want to generate new models or results you can explore already computed results by right clicking on the nodes and choosing the output table for display. If you want to make modifications to the workflow you can click on the nodes and change their settings. Some example changes could be: - Change input data by clicking on File Reader nodes \(positive or negative) - Change number of iterations for loop by going into Loop x-times meta-node and clicking on Counting Loop Start then setting the number - Changing sampling ratio by going into Loop x-times meta-node and in sampling meta-node changing partitioning node’s settings - Using your desired feature set; go into Loop x-times/studies. If you want to add another study with different feature set copy paste one of the meta-nodes \(e.g. Ng) connect it in the same way as the existing ones. Right click and select Reconfigure to change meta-node name. Then go into your meta-node select filter \(feature selection). Inside that meta-node, there are two column filter nodes; one for learning another for testing data, in these nodes select your choice of features. In classifier and CombineLearnedStats meta-nodes you should do renaming since they would be set to Ng in this instance. Prediction The prediction workflow requires a column named "Accession" for joining. If your data has no such columns you can use RowID node to create unique accession values. Figures Figure 1. Overall training workflow Figure 2. MCCV and model generation Figure 3. Sampling. Figure 4.

Studies \(feature groups). Figure 5. Feature selection and application of 3 classifiers. Figure 6. Model sorting, selection and saving as PMML files. Figure 7. Prediction workflow. Figure 8 Prediction Meta-node Figure 9 Decision Tree/Naïve Bayes Meta-node Figure 10 Consensus Result Meta-node Figure 11 Consensus Model Meta-node Figure 12 Visualization Meta-node

Timing

-

Troubleshooting

-

Anticipated Results

-

References

-

Acknowledgements

-

Figures

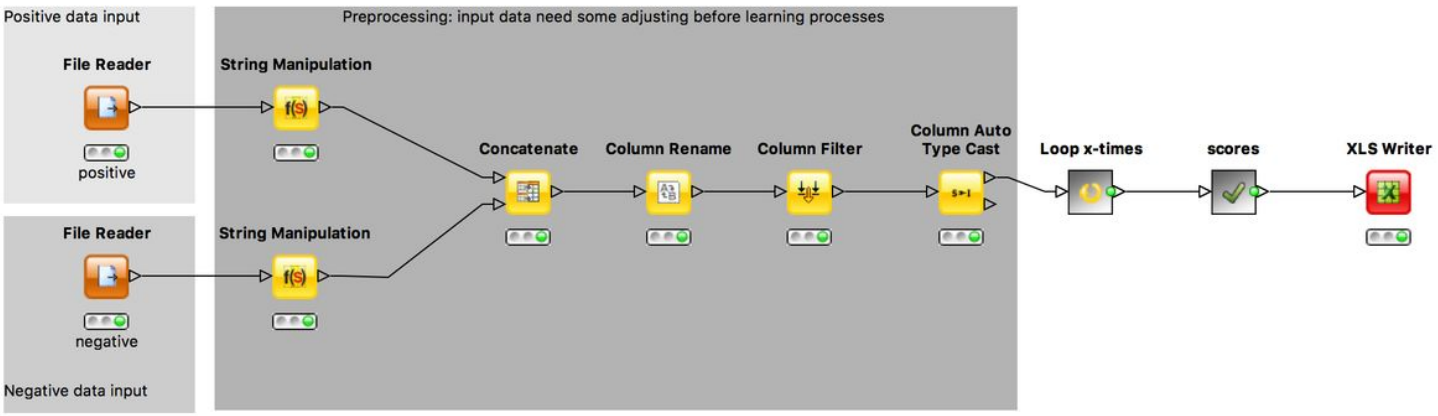

Figure 1

Overall training workflow Overview of the training workflow. Two file input nodes on the left (negative and positive data) channel the data to preprocessing (dark gray box). Afterwards training is done in Loop-x-times and scores are stored.

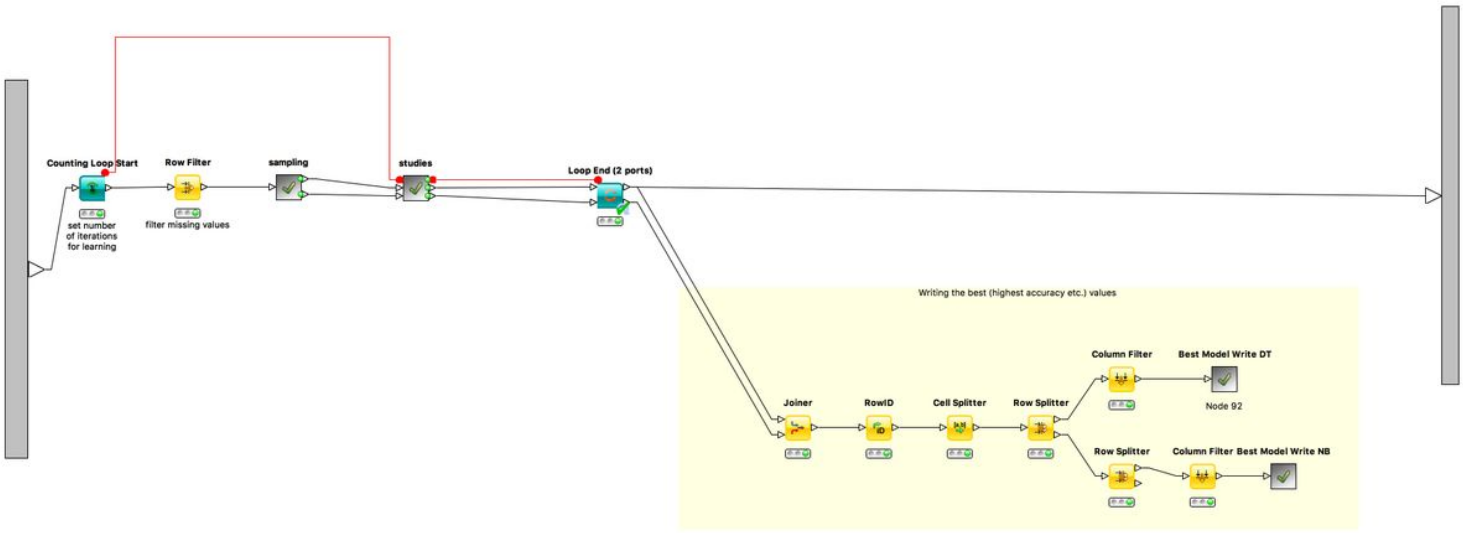

Figure 2

MCCV and model generation Zoom into the loop-x-times meta node from Figure 1. Monte Carlo cross validation scheme for training and testing during model generation. In the yellow box the best model is determined and stored. All results are aggregated and available as output from the meta node.

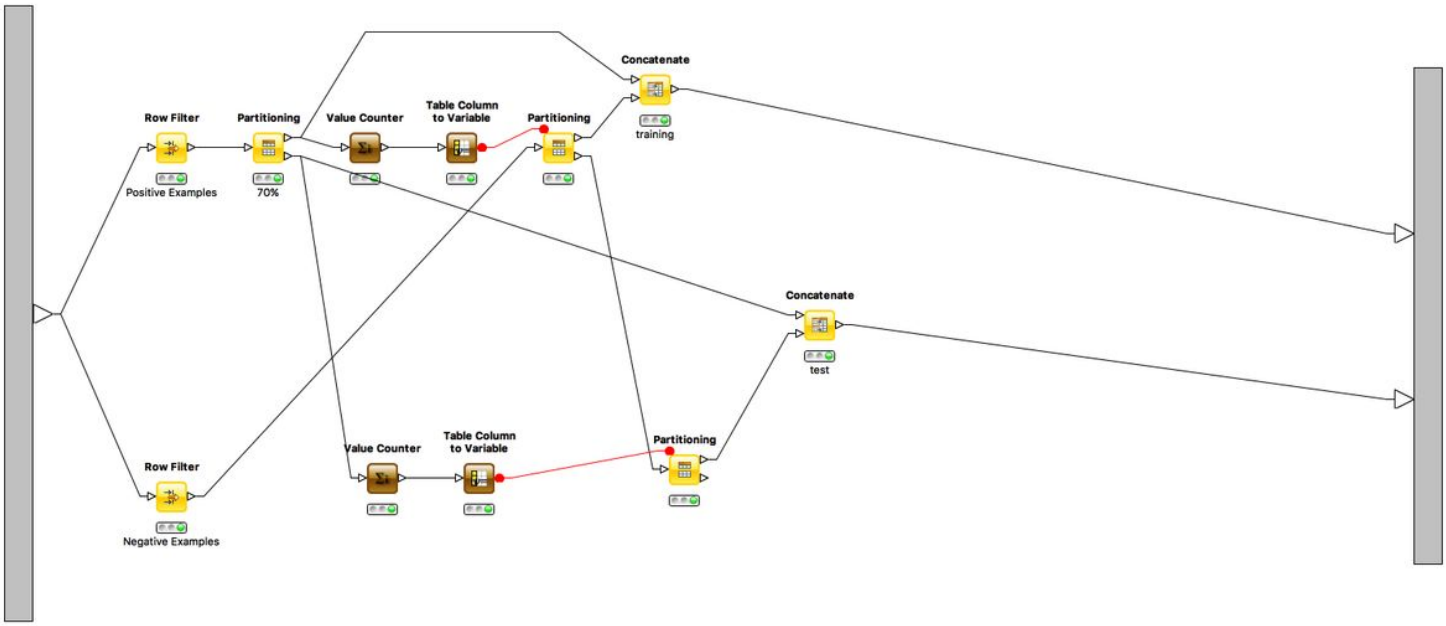

Figure 3

Data Sampling Incoming mixed data is split into positive and negative examples and then randomly sampled with equal amount of positive and negative examples. These are split into training and testing

examples and are connected to the output of the meta node.

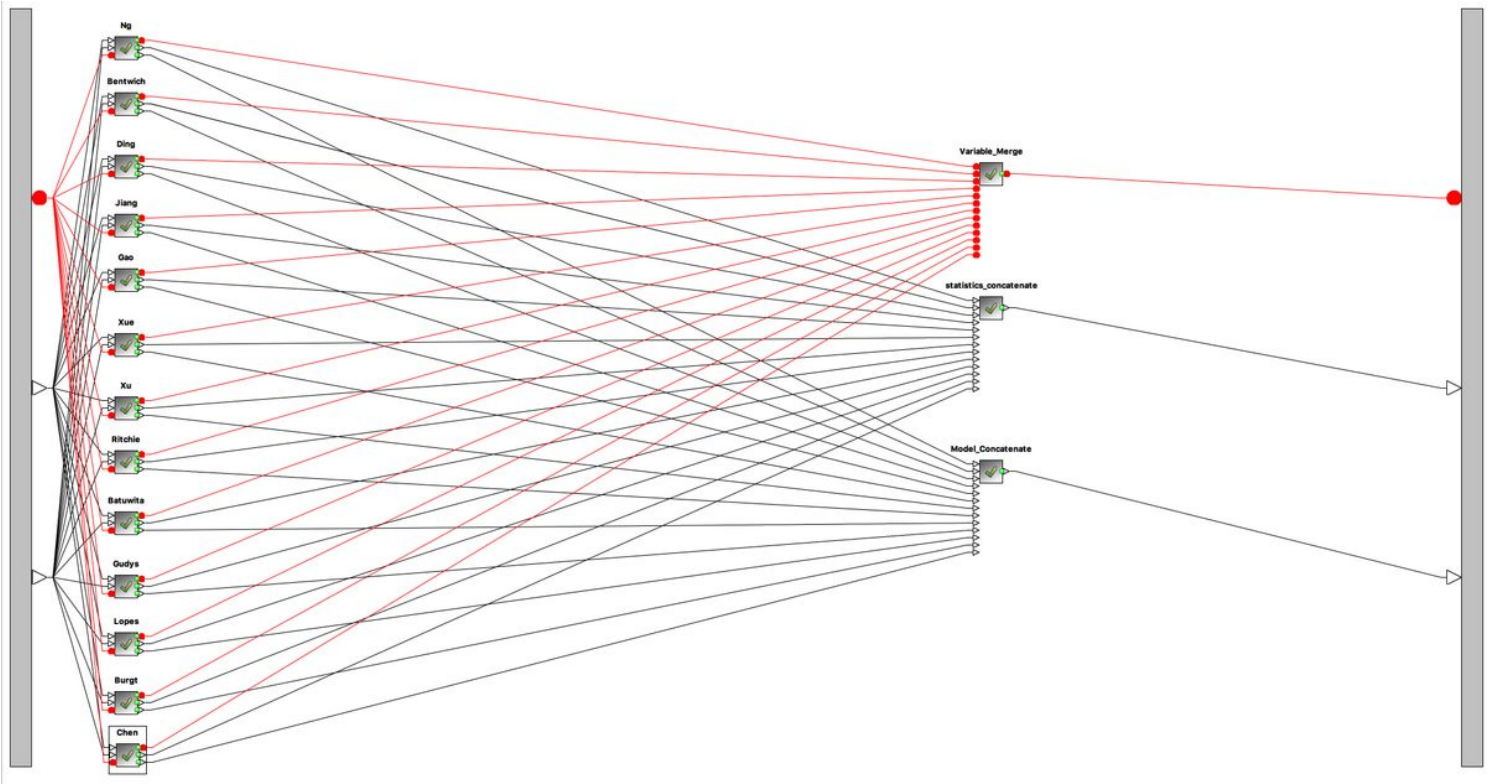

Figure 4

Training According to Studies Thirteen previous studies' feature sets are used to create different models which are exported from this meta node along with statistics.

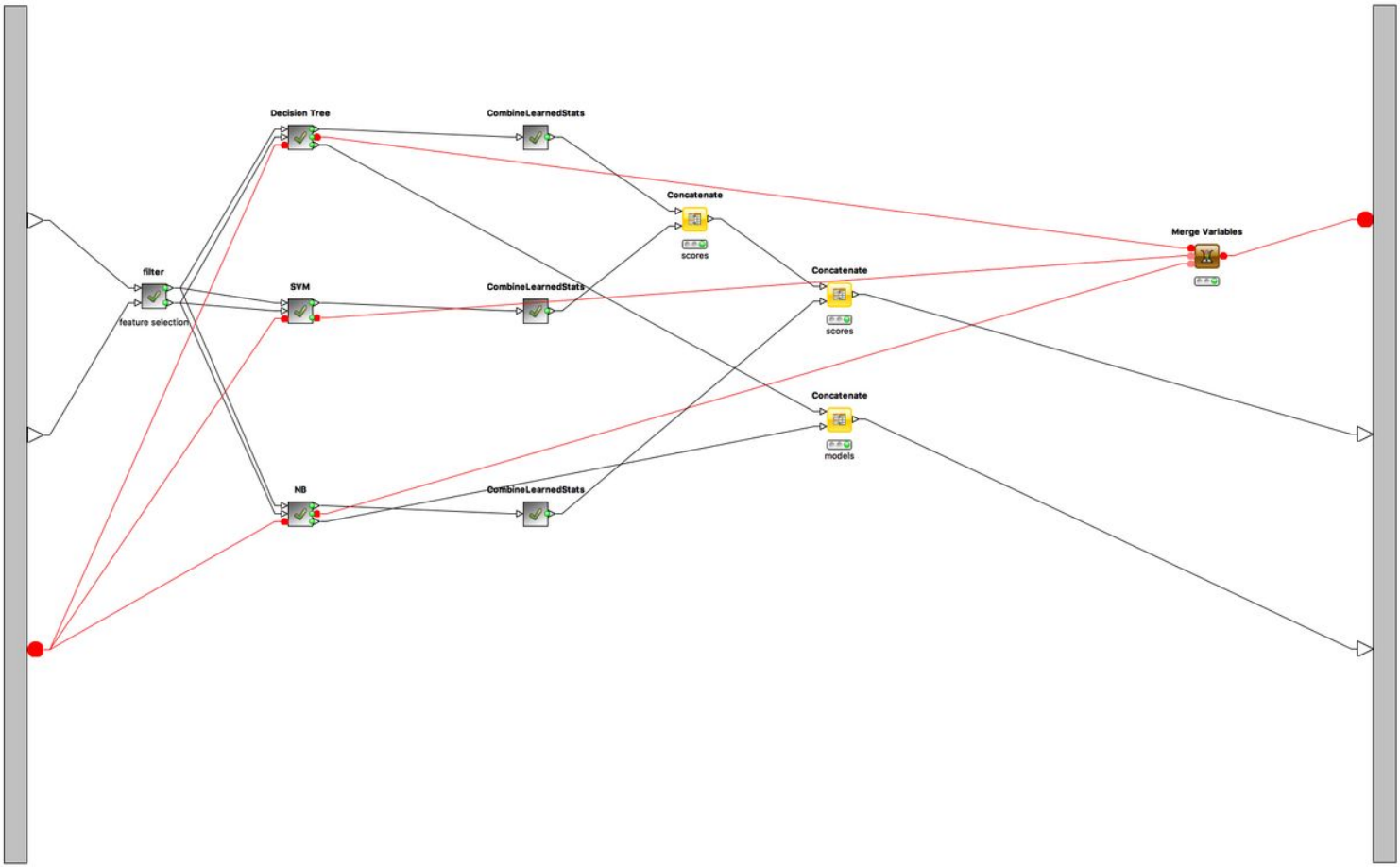

Figure 5

Feature Selection and Application of 3 Classifiers In order to train models for the 13 studies (Figure 4) the features are first filtered on a per study basis and then submitted to three learners (here: NB, SVM, and decision tree). Results are collected and forwarded to the out port of the meta node.

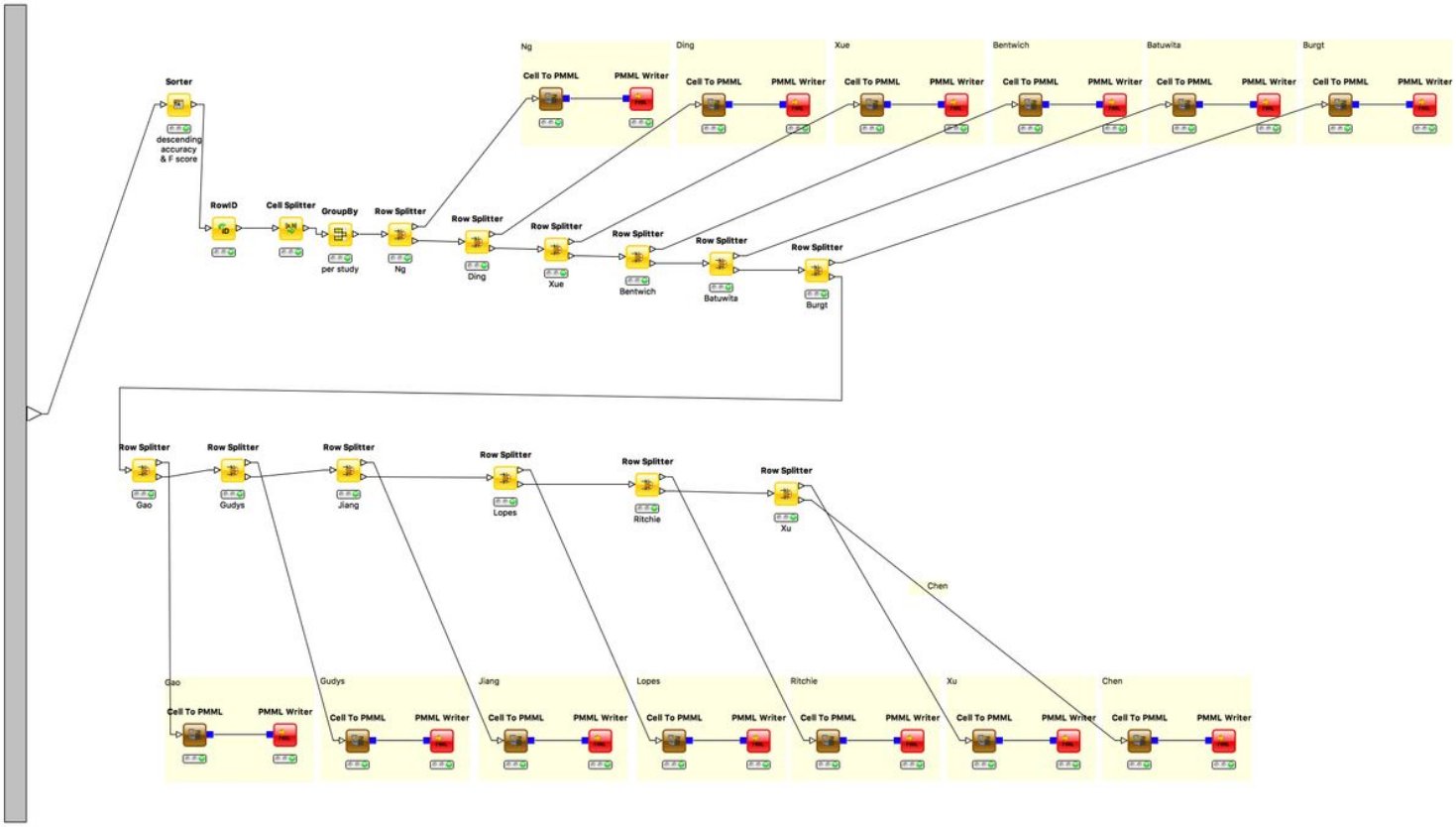

Figure 6

Model Sorting, Selection, and Storing as PMML Model This meta node is a sink which evaluates all models that were created and stores the best as PMML. These models are stored such that they are automatically used during the prediction process. Models are stored on the file system and no output is provided from the meta node.

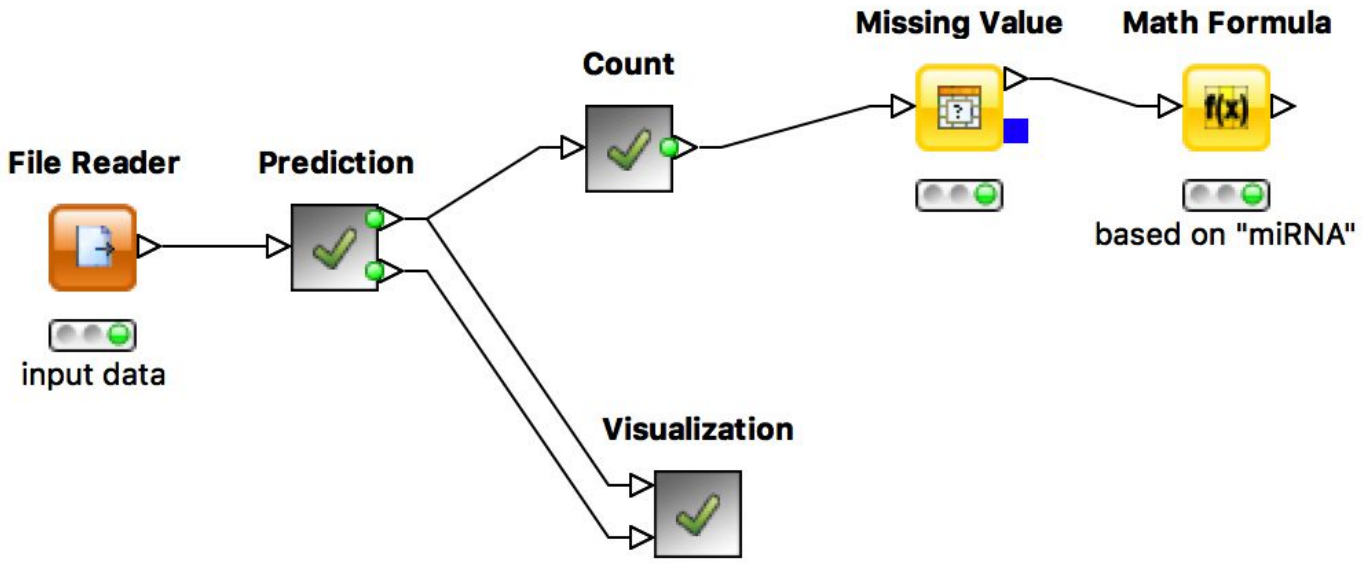

Figure 7

Prediction Workflow Data is read in and the various PMML models (see Figure 6) are invoked for prediction. Some statistics and visualizations are produced in the associated meta nodes 'Count' and 'Visualization'.

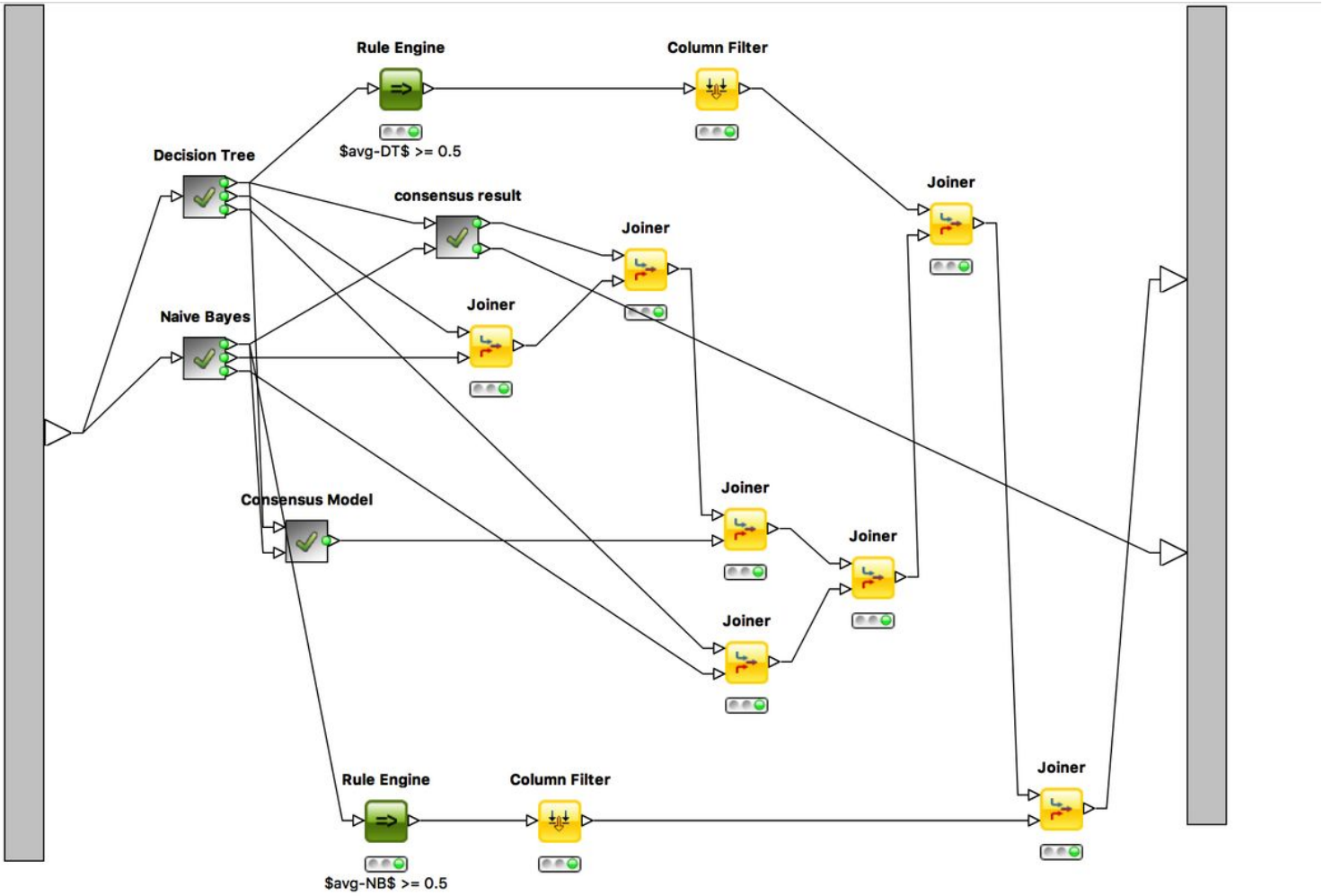

Figure 8

Prediction Meta-Node All 13 established decision tree and naive Bayes PMML models are applied to the data. The results are made available on a per study basis and as differnt consensus models. Results are forwarded to the meta node's out port.

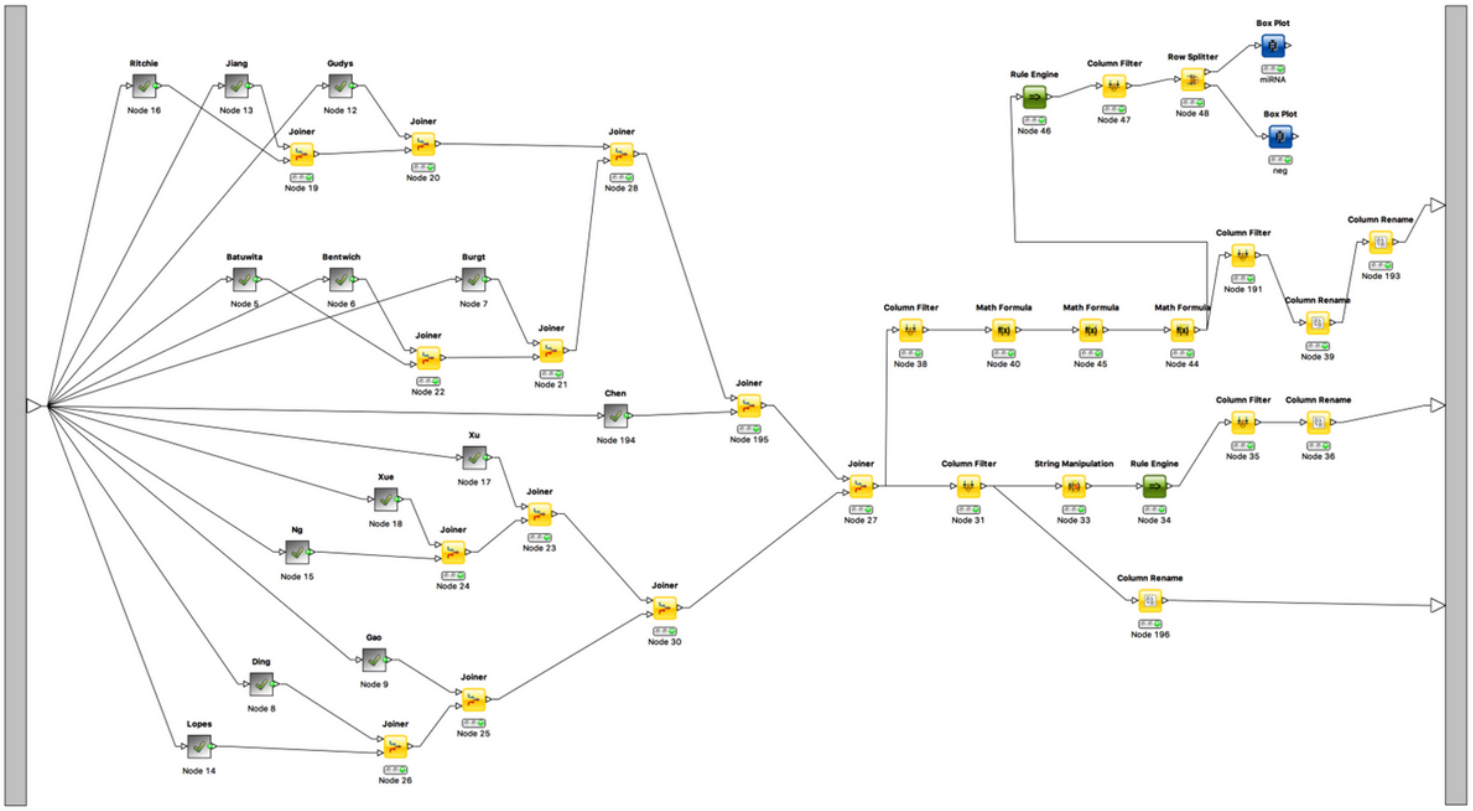

Figure 9

Decision Tree and Naive Bayes Meta-Node PMML models produced for the 13 studies are applied in their associated meta-nodes and results are forwarded to the meta-node's out port.

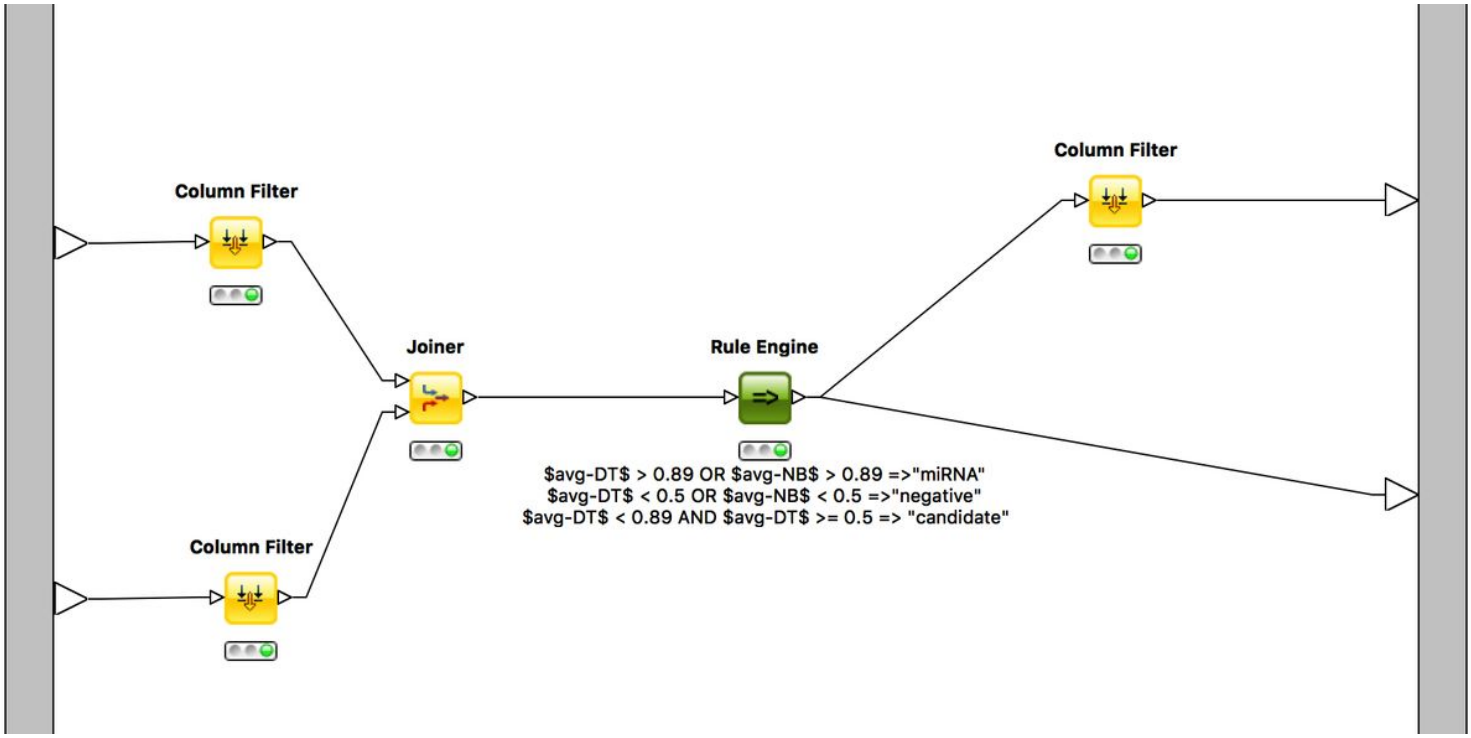

Figure 10

Consensus Result Prediction Meta-Node One of the consensus models which takes as input the average decision tree and average naive Bayes results and decides whether the example is a miRNA or not based on some rules.

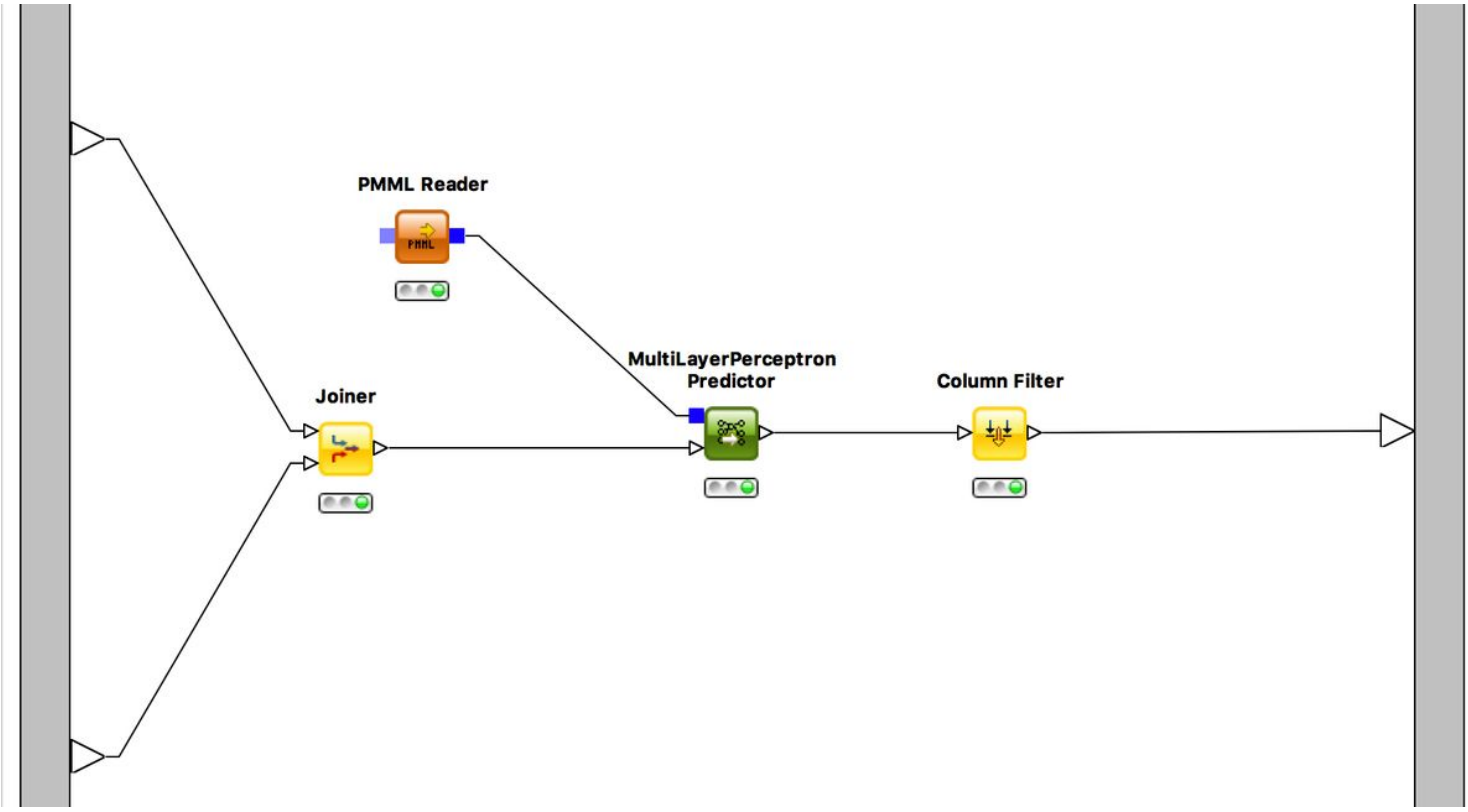

Figure 11

Consensus Model Prediction Meta-Node Similar to Figure 10 a consensus model calculation is presented. In this case a multi layer perceptron was trained and its PMML model was stored during training. This model is applied here.

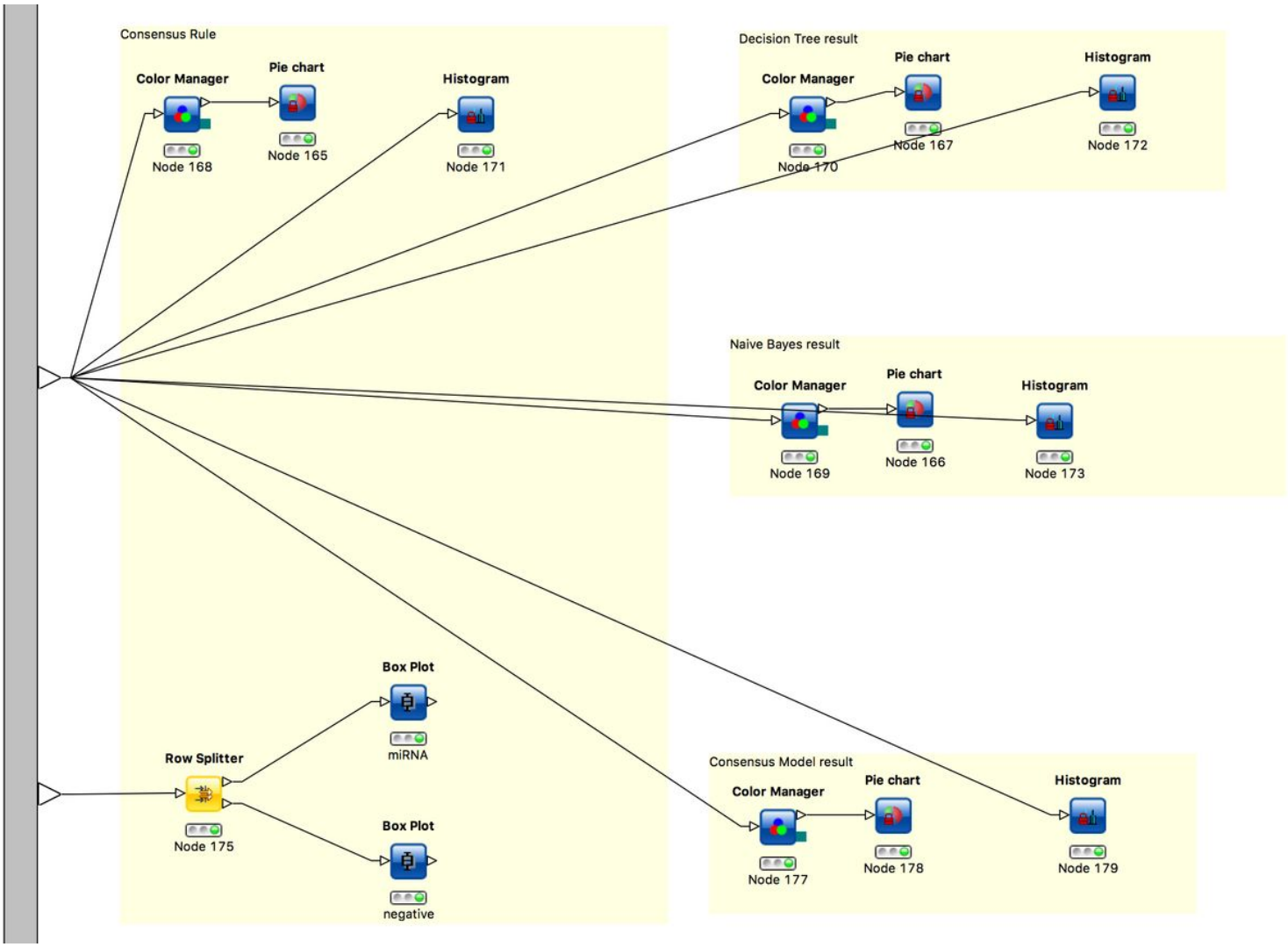

Figure 12

Visualization Meta-node Various visualization of statistics and results from training and testing as well as application of the models.