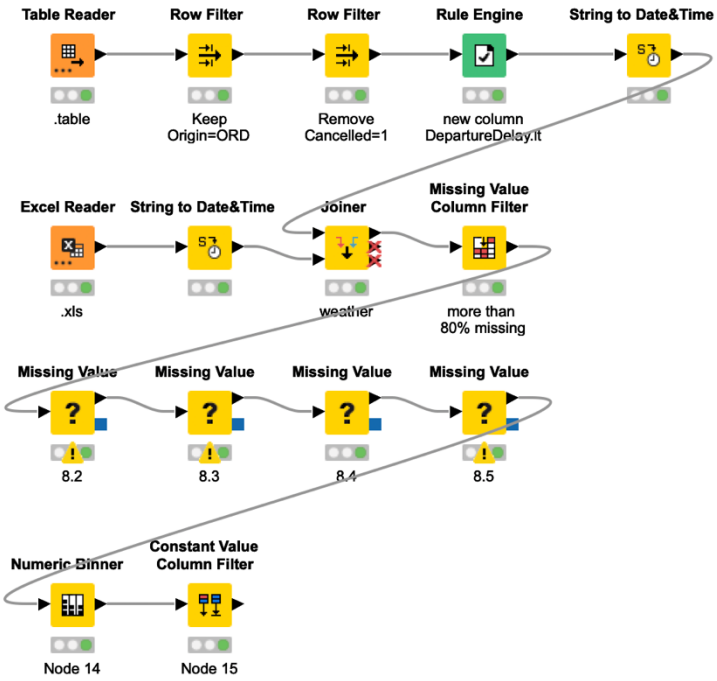
1. Exercise Data Import and Pre-Processing

Goal: Access, extend and prepare data.



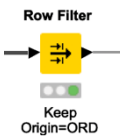
Download Datasets from Moodle:

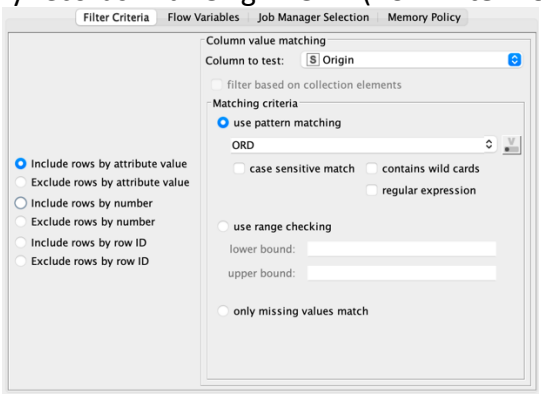
1. *AirlineDataset.table*
2. *GHCN-Daily_source.xls contains daily weather information like precipitation, snowfall, snow depth, temperature, wind speed and wind direction measured at Chicago O'Hare International Airport.*

1. Read the data *AirlineDataset.table* ***(Table Reader node)***

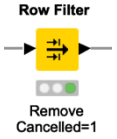
Advanced Settings → nichts angekreuzt

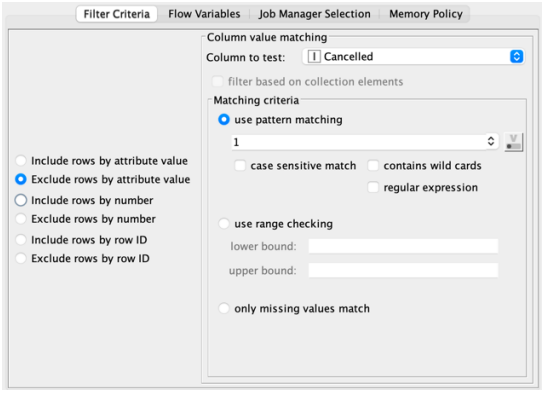
2. Keep only records with Origin=ORD (**Row Filter node)**



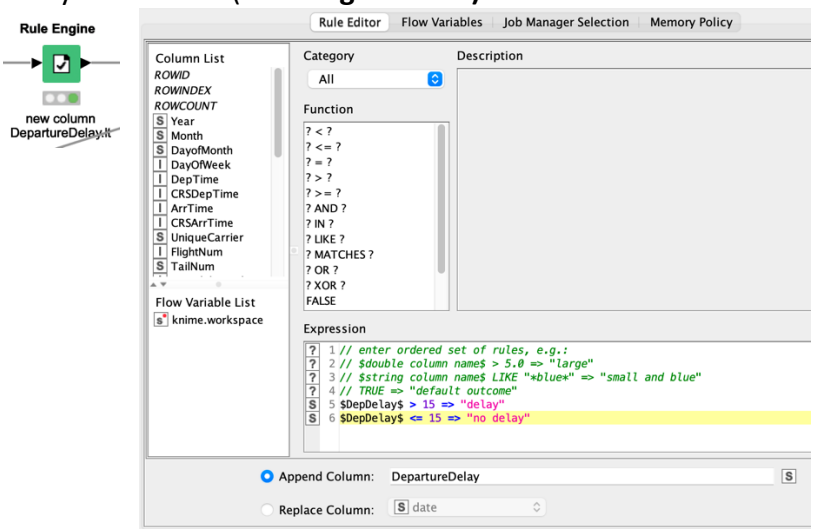


3. Remove canceled flights, i.e. Cancelled=1 (**Row Filter node)**

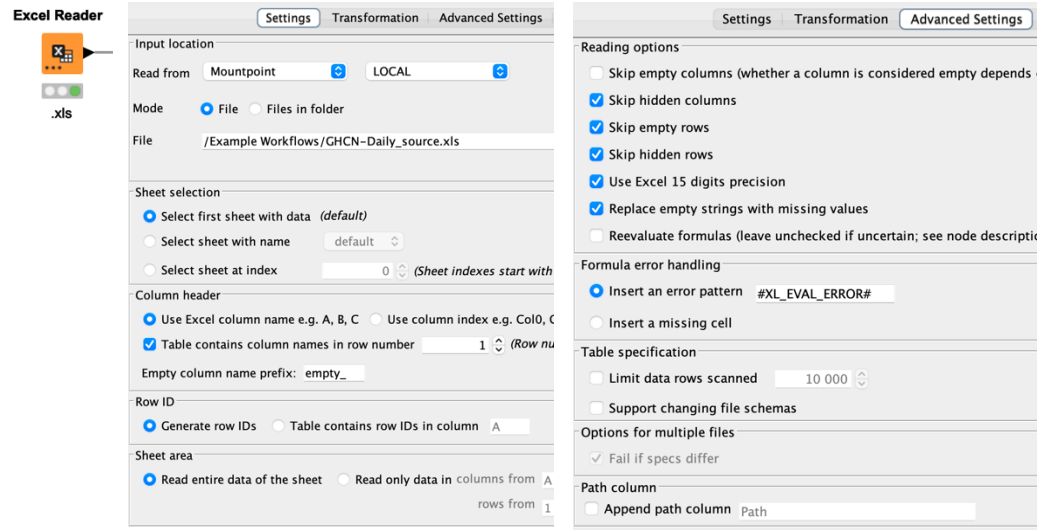




4. Create a new column called DepartureDelay. It acquires the value "delay" if DepDelay > 15min and "no delay" otherwise (**Rule Engine node).**



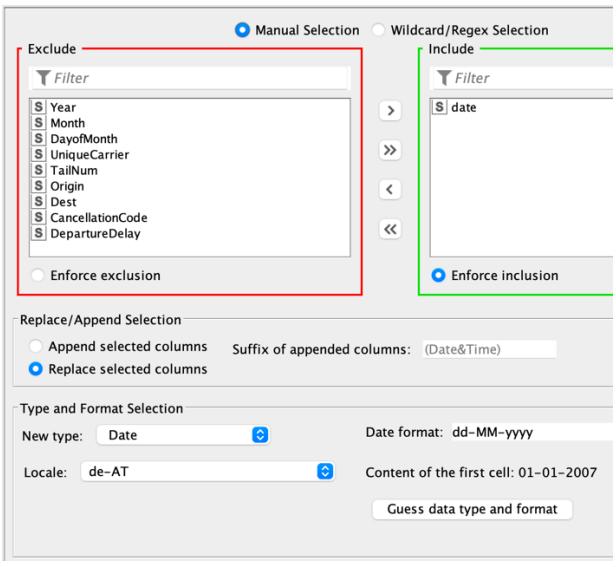
5. Read the *GHCN-Daily_source.xls file* (***Excel Reader (XLS) node)***



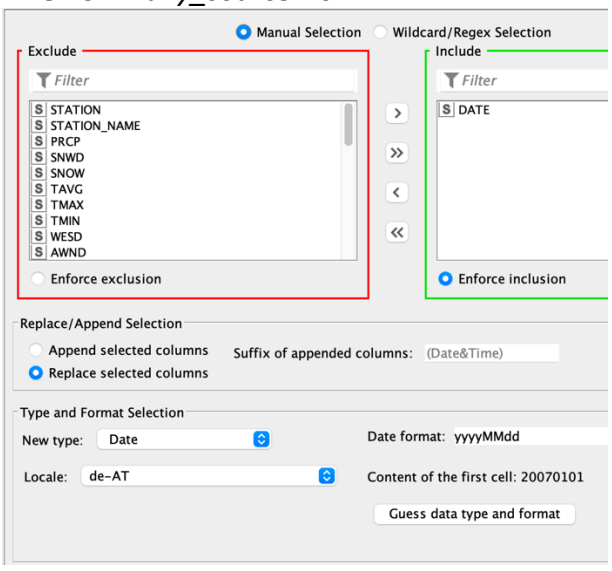
6. Convert the date columns in both datasets from the data type string to date&time (**String to Date&Time node)**

String to Date&Time

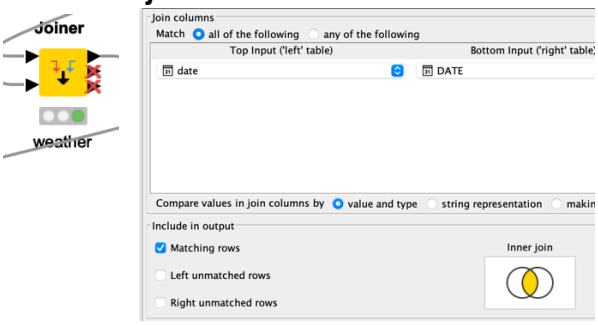
AirlineDataset.table



GHCN-Daily_source.xls

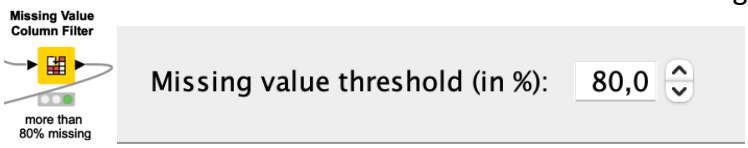


7. Join the weather data with the airline data using the date columns as the joining columns (**Joiner node). Use inner join.**



8. Handle missing values:

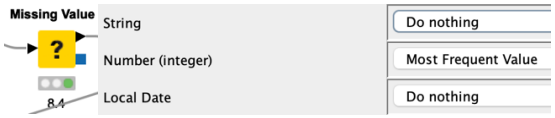
8.1 Remove columns that contain more than 80% missing values (**Missing Value Column Filter node)**



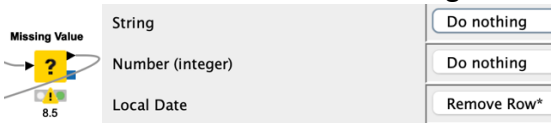
8.2 If the data contains rows where the value of DepDelay is missing, remove them (**Row Filter node OR Missing Value node)**

8.3 Set missing values in string columns to a fixed value "unknown" (**Missing Value node)**

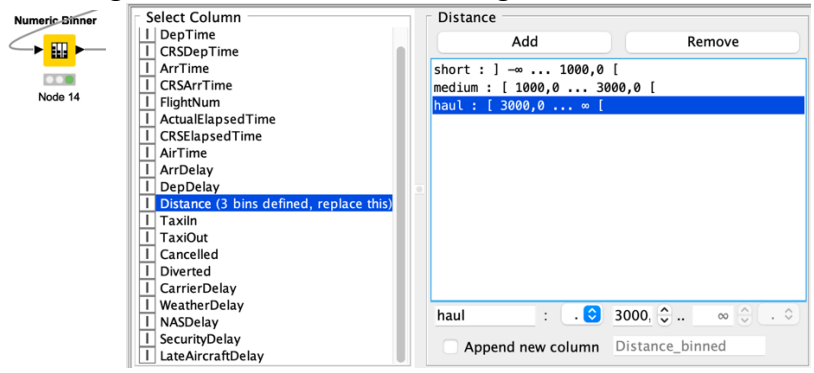
8.4 Set missing values in integer columns to the most frequent value in the column (**Missing Value node)**



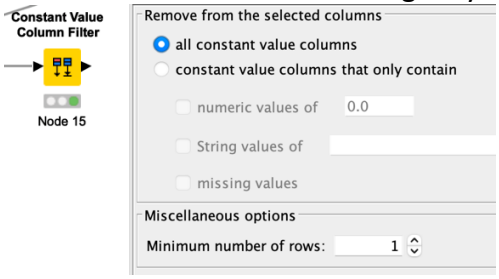
8.5 Remove rows that have missing values in a column of type date&time (**Missing Value node)**



9. Bin flight distance to three bins: long, short and medium haul (**Numeric Binner node)**

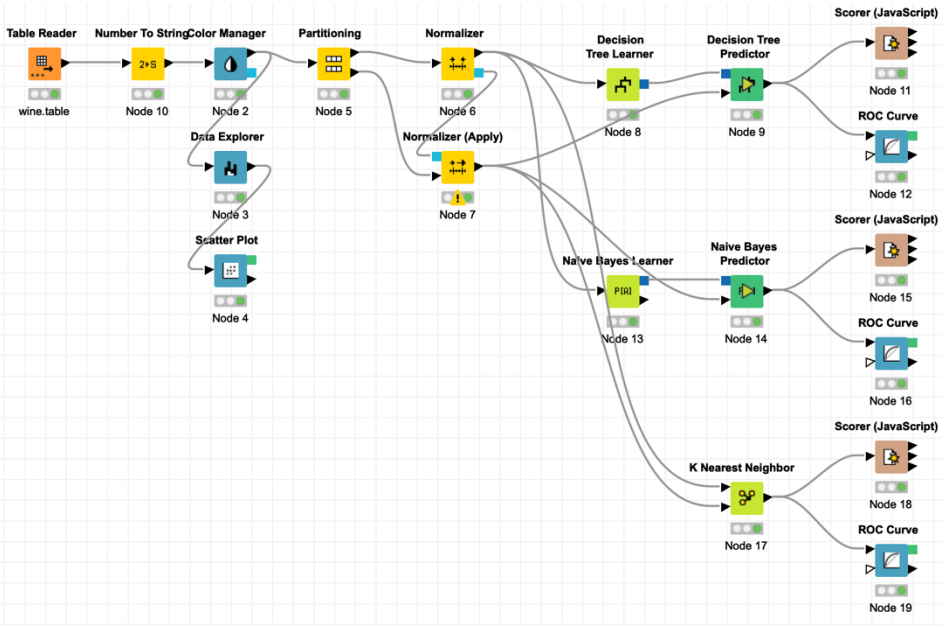


10. Remove columns containing only constant values (**Constant Value Column Filter node)**



2. Wine data classification exercise

-Chemical properties of 178 wines are examined, resulting in 13 numerical features.
-There are 3 different types of wines in this data set, described by the column Type.
-Goal of this analysis to classify these wines based on their features.

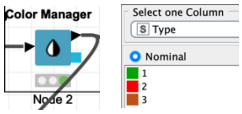


1. Reading the data set
-Read the file "*wine.table*" with the **Table Reader node**

Advanced Settings → nichts angekreuzt

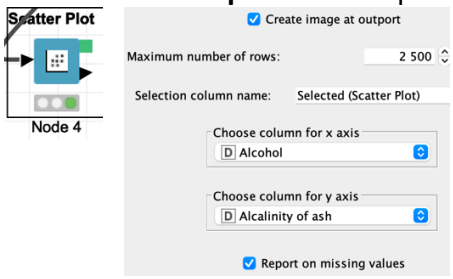
-**Number to String node**

2. Explore the data
-Use the **Color Manager node** to assign colors to different classes of the target variable Type

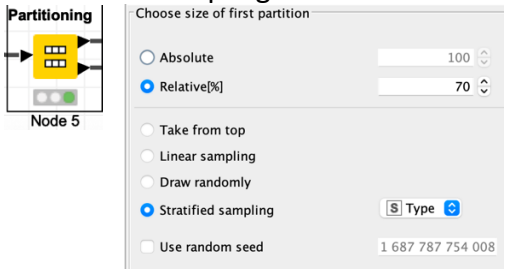


-Use the **Data Explorer node** to examine statsitics and distributions of the features

-Use the **Scatterplot node** to plot various attributes

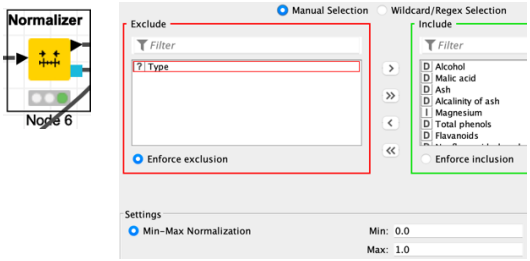


3. Partitioning
-Use the **Partitioning node** to split the data set into the training (70%) & testing (30%) data sets
-Stratified sampling

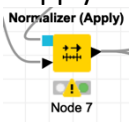


4. Normalization

-For the training data, normalize numerical features to the range of [0,1] with the **Normalizer node**



-Apply the normalization from the training data to the testing data with the **Normalizer (Apply) node**



5. Train and apply a decision tree classification model

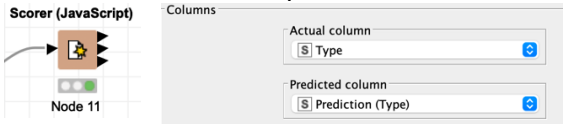
-Train a decision tree model with the **Decision Tree Learner node**.

-Apply the trained model to the testing data with **Decision Tree Predictor**

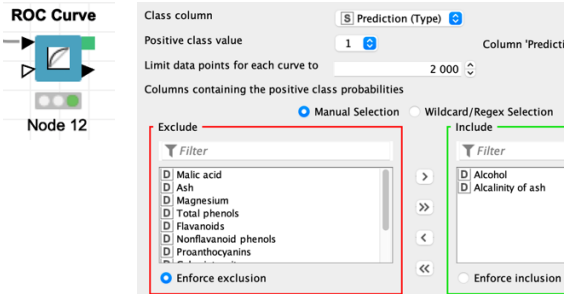
-Make sure to output class probabilities

???

-Evaluate the model performance with the **Scorer (JavaScript) node**



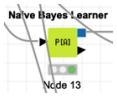
-Plot the ROC curve



-Adjust parameters of the Decision Tree Learner to improve the classifier performance

6. Train and apply a Naive Bayes classification model

-Train a naive Bayes model with the **Naive Bayes Learner node**.

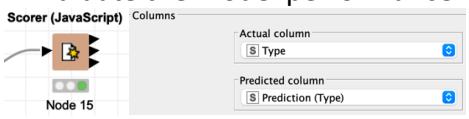


-Apply the trained model to the testing data with **Naive Bayes Predictor**

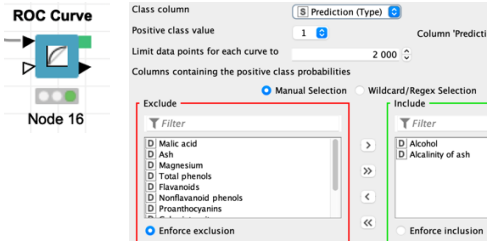
-Make sure to output class probabilities

???

-Evaluate the model performance with the **Scorer (JavaScript) node**

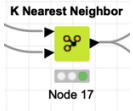


-Plot the **ROC curve**

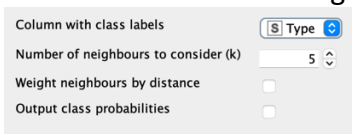


7. kNN classification model

-Apply the kNN classification model with the **K Nearest Neighbor node**



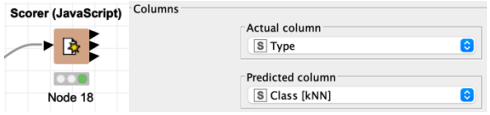
-Set k=5. Use the training data set as the model.



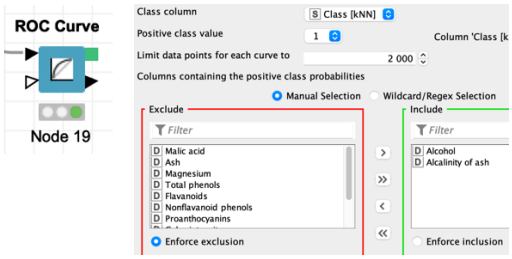
-Make sure to output class probabilities

???

-Evaluate the model performance with the **Scorer (JavaScript) node**



-Plot the ROC curve



-Adjust parameters of the K Nearest Neighbor node to improve the classifier performance