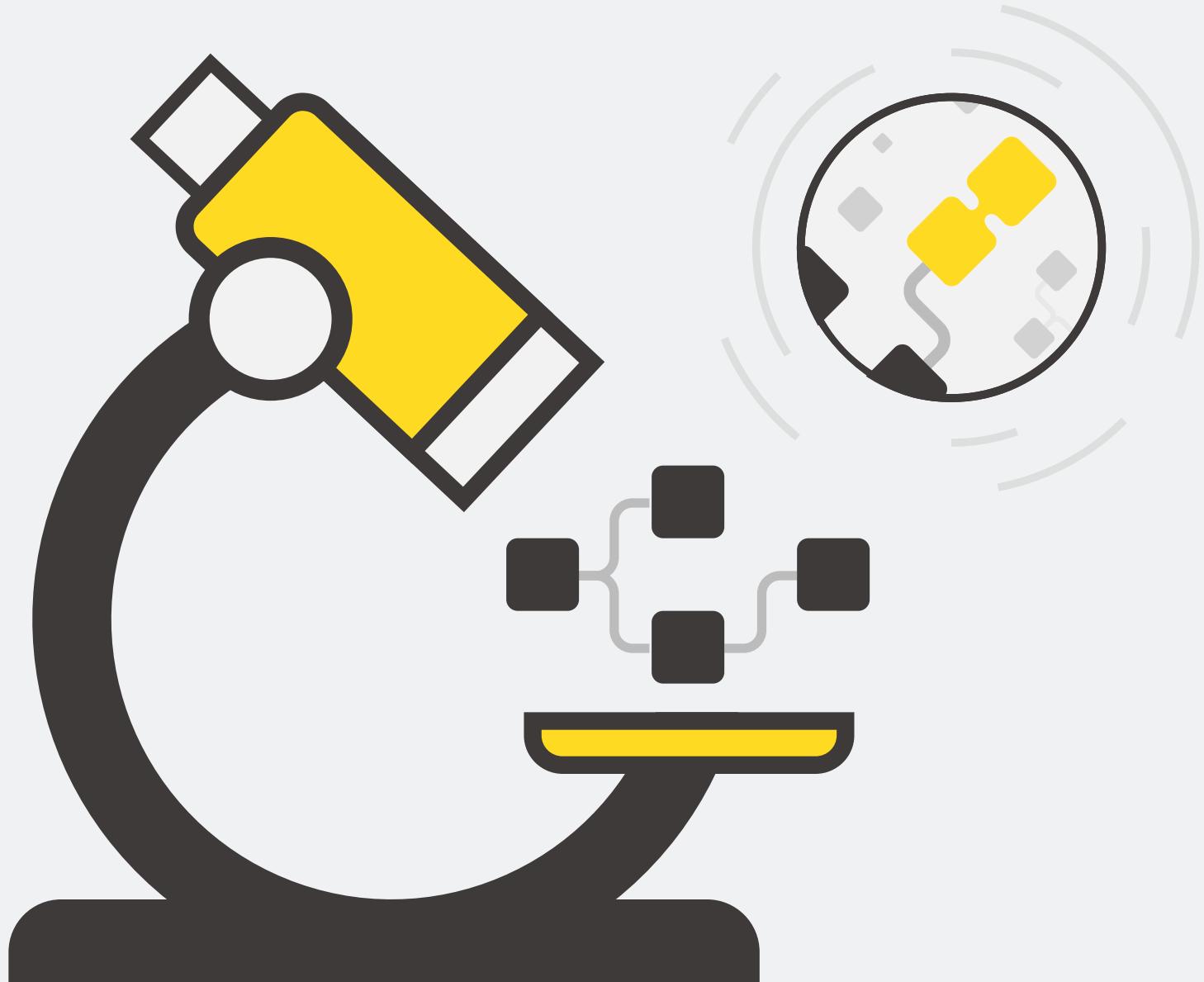


# KNIME for Life Sciences

A Collection of Use Cases



Copyright©2021 by KNIME Press

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording or likewise.

This book is based on **KNIME 4.4**.

For information regarding permissions and sales, write to:

KNIME Press  
Hardturmstrasse 66  
8005 Zurich  
Switzerland

[knimepress@knime.com](mailto:knimepress@knime.com)

# Table of Contents

|   |    |
|---|----|
| Introduction.....   | 7  |
| Chapter 1: Cheminformatics.....   | 8  |
| 1.1    Tutorials for Computer Aided Drug Design using KNIME workflows .....   | 9  |
| Workflow 1: Acquire compound data from ChEMBL .....   | 10 |
| Workflow 2: Filter datasets by ADME criteria .....  | 11 |
| Workflow 3: Set alerts based on unwanted substructures.....   | 12 |
| Workflow 4: Screen compounds by compound similarity .....   | 13 |
| Workflow 5: Group compounds by similarity.....  | 14 |
| Workflow 6: Find the maximum common substructure in a collection of compounds .....   | 15 |
| Workflow 7: Screen compounds using machine learning methods .....   | 16 |
| Workflow 8: Acquire structural data from PDB .....  | 17 |
| Requirements.....   | 17 |
| References.....   | 17 |
| 1.2    Training a machine learning model - to building a predictive web application in three steps                            | 18 |
| 1. Model selection and parameter optimization.....  | 18 |
| Hyperparameter optimization .....   | 19 |
| Compare Model Performances.....   | 20 |
| Report on Best Hyperparameter and Model Performance.....  | 21 |
| 2. Integrated Deployment of the best performing model .....   | 22 |
| 3. Creating a web application .....   | 24 |
| Wrapping up .....   | 26 |
| 1.3    Multitasking doesn't always make things worse: interactive bioactivity prediction with multitask neural networks ..... | 28 |
| Loading the network and generating predictions .....  | 28 |
| Showing the predictions in an interactive heatmap .....   | 30 |
| Comparing predictions to measured values.....   | 32 |
| Wrapping Up.....  | 34 |
| Chapter 2: Bioinformatics .....   | 35 |
| 2.1    Analyzing Gene Expression Data with KNIME.....   | 36 |
| Express Yourself! .....   | 36 |

|  |    |
|--|----|
| Gene Expression Analysis .....   | 36 |
| Transcription .....  | 36 |
| Input data.....  | 37 |
| Find differentially expressed genes.....   | 38 |
| View of differentially expressed genes .....   | 39 |
| Clustering.....  | 40 |
| Pathway enrichment .....   | 41 |
| View compounds targeting gene product of interest.....                                     | 42 |
| Summary.....   | 43 |
| References.....  | 44 |
| 2.2    Gut Microbiome Analysis with KNIME Analytics Platform.....                          | 45 |
| Human Gut Microbiome .....   | 45 |
| The 16S ribosomal RNA (16S rRNA) gene .....  | 46 |
| A KNIME Workflow for Gut Microbiome Analysis of IBD Patients from 16S sequencing data ...  | 46 |
| 1. Download FASTQ sequences from ENA .....   | 47 |
| 2. Create an Amplicon Sequence Variant (ASV) table.....                                    | 48 |
| 3. Create taxonomic profile at a desired taxonomic rank.....                               | 52 |
| 4. Visualize the results.....  | 53 |
| Summary.....   | 56 |
| References.....  | 56 |
| 2.3    Variant Prioritization - Reproducible Workflow with Domain Expert Interaction ..... | 58 |
| Variant Prioritization.....  | 58 |
| Variant Prioritization Workflow .....  | 59 |
| Convert coordinates from one assembly to another.....                                      | 62 |
| Retrieve variant annotations from Variant Effect Predictor (VEP) .....                     | 62 |
| Summary .....  | 65 |
| References.....  | 65 |
| Chapter 3: Text Mining .....   | 67 |
| 3.1    Predicting the Purpose of a Drug .....  | 68 |
| Gathering drug names and related articles.....   | 69 |
| Dictionary creation (Drug names).....  | 70 |
| ATC Classification System .....  | 70 |

|  |     |
|--|-----|
| Corpus creation .....  | 70  |
| Preprocessing, model training and evaluation.....                                  | 70  |
| Create a co-occurrence network and predict drug purposes .....                     | 72  |
| Extract interesting subgraphs .....  | 74  |
| Summary.....   | 75  |
| References.....  | 75  |
| 3.2 Will They Blend? KNIME meets the Semantic Web.....                             | 76  |
| The Challenge.....   | 76  |
| The Experiment .....   | 77  |
| Visualization .....  | 79  |
| 3.3 Exploring a Chemistry Ontology with KNIME .....                                | 82  |
| ChEBI.....   | 82  |
| Let's start! .....   | 83  |
| Wrapping up .....  | 90  |
| References.....  | 91  |
| Chapter 4: Lab Data .....  | 92  |
| 4.1 Near Infrared Spectroscopy (NIR) Data Analysis using KNIME .....               | 93  |
| 1. NIR Spectroscopy data pre-treatment.....  | 93  |
| 2. Visualization, Clustering, and PCA analysis of Preprocessed Spectral Data ..... | 99  |
| 3. Similarity Search Using Inhouse Database .....                                  | 103 |
| Conclusion.....  | 107 |
| 4.2 User-friendly End-to-End Lab Automation in Action.....                         | 108 |
| Standardized lab automation with KNIME Analytics Platform.....                     | 108 |
| SiLA for open connectivity in lab automation .....                                 | 108 |
| How does SiLA integrate with KNIME?.....   | 109 |
| Use Case: Automatic Analysis of Microplate Data.....                               | 111 |
| Use Case: Automatic Analysis of Imaging Data .....                                 | 111 |
| Summary.....   | 113 |
| References.....  | 113 |
| 4.3 What are the FAIR guiding principles and how to FAIRify your data .....        | 114 |
| What are FAIR data? .....  | 114 |
| How you can use KNIME to help make your data FAIR.....                             | 115 |

|   |     |
|---|-----|
| Summary of how KNIME contributed to FAIR data management..... | 121 |
| References.....   | 122 |
| All References .....  | 123 |
| Index .....   | 126 |

# Introduction

As in every other field, data plays a big role in the life sciences too. New technologies and developments have boosted the possibilities to generate large amounts of data, while analysis of the increasing amount of data and big data is gaining in importance. This spans all kind of life science fields.

This not only holds true for the field of bioinformatics, where rapid advances in sequencing technology creates huge amount of data, driving personalized medicine. In-silico techniques also play an important role in the field of cheminformatics: More and more screening data is available and computer-aided drug design is widely used in the pharma industry. Machine-assisted data gathering and analysis is also increasing in relevance when it comes to lab set-up, where processes will be increasingly automatized in future.

All these developments also reflect on the amount of biomedical literature published – growing year on year. Text mining aims to automatically distill information, extract facts, discover implicit links, and generate hypotheses relevant to user needs.

With this book we want to give an overview of pertinent cases from different fields of the life sciences, in specific cheminformatics, bioinformatics, text mining and laboratory data. Each chapter is dedicated to one of these fields and covers three use cases. At the beginning of each chapter, concise abstracts describe each use case to allow readers to quickly find the cases that are applicable to them.

All examples described in this book refer to at least one workflow, which are available for readers to download from the KNIME Hub. Workflow links are provided at the beginning of each section. We will update this book as new, interesting use cases arise.

*Alice Krebs, Editor*

# Chapter 1: Cheminformatics

In this chapter we want to highlight three workflows and use cases from the cheminformatics field.

## **Tutorials for Computer Aided Drug Design using KNIME workflows**

This collection of interconnected workflows demonstrates eight common tasks in computer-aided drug design. They are available as individual workflows, but also as one large workflow that combines all the steps. It is illustrated using the epidermal growth factor receptor (EGFR), but can easily be applied to other targets of interest. Individual steps and tasks comprise obtaining data from the ChEMBL web services, applying the Lipinski's rule of five and PAINS filter, screening compounds for similarity based on fingerprints, hierarchical clustering, finding maximum common substructure, using machine learning methods to predict compound activity and fetching structural data automatically from the Protein Data Bank (PDB). All workflows require only KNIME AP and integrate functionality of the RDKit und Vernalis extensions.

## **Training a machine learning model - to building a predictive web application in three steps**

This story demonstrates how to deploy an optimized machine learning model by either integrated deployment or as a web application. We took the hyperparameter optimization workflow of a previous story as a starting point and did some adjustments by introducing components that allow the user to interact and make choices. We also introduce integrated deployment, which allows us to capture the parts of the workflow that are needed for the production workflow. This requires only KNIME version 4.2 or higher. Furthermore, we describe how to create an interactive web application based on the workflow, where users can enter their data and get a model prediction and with some visualization. This however requires a KNIME Server.

## **Multitasking doesn't always make things worse: interactive bioactivity prediction with multitask neural networks**

In this story we use a multitask neural network model to predict the bioactivity of input molecules. The model has been trained and validated externally with Jupyter notebooks using the PyTorch framework. The model was then imported into KNIME via the ONNX format and converted to a Tensorflow network. It predicts the activity of the 12 input molecules on 560 targets based on their fingerprints that were generated with the RDKit extension. Any molecules in SMILES format and a molecule identifier (e.g. name or ID) can serve as input. In parallel, we also retrieve experimental data from ChEMBL if present. We investigate the results with two interactive component visualizations: first a heatmap of the predicted bioactivities, and secondly with a view that displays the predictions and measurements for comparison.

# 1.1 Tutorials for Computer Aided Drug Design using KNIME workflows

By Andrea Volkamer, Jaime Rodriguez-Guerra & Dominique Sydow

Find the workflow(s) here: <https://kni.me/s/i6jmSEXRH0OMF8R>

Jupyter Notebooks offer an incredible potential to disseminate technical knowledge thanks to its integrated text plus live code interface. This is a great way of understanding how specific tasks in the Computer-Aided Drug Design (CADD) world are performed, but only if you have basic coding expertise. While users without a programming background can simply execute the code blocks blindly, this rarely provides any useful feedback on how a particular pipeline works. Fortunately, more visual alternatives like KNIME workflows are better suited for this kind of audience.

Here we want to introduce our new collection of tutorials for computer-aided drug design (Sydow and Wichmann et al., 2019). Building on our Notebook-based [TeachOpenCADD](#) platform (Sydow et al., 2019), our TeachOpenCADDKNIME pipeline consists of eight interconnected workflows (W1-8), each containing one topic in computer-aided drug design.

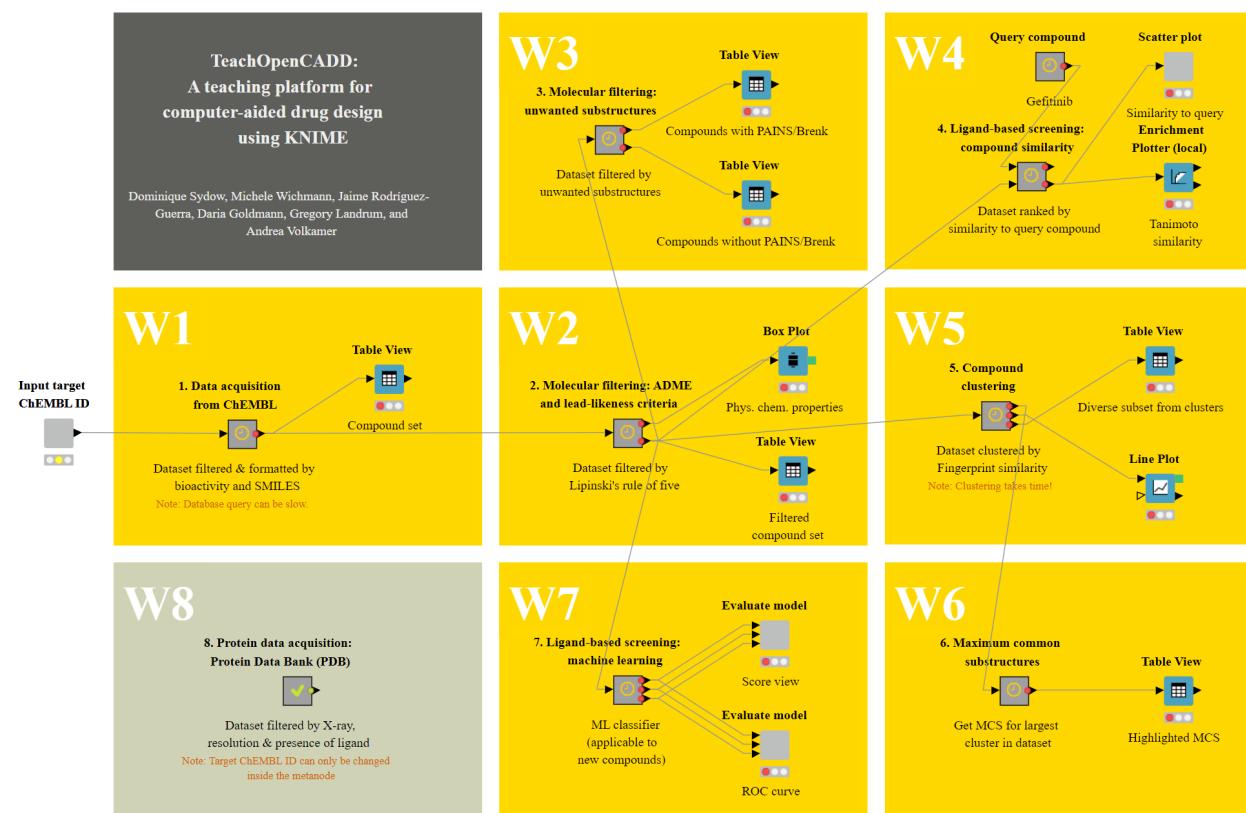


Figure 1 The visual capabilities of the KNIME Platform are evident. This is not a diagram of the TeachOpenCADD KNIME workflows, but the actual project as rendered in KNIME itself. Each box can be accessed individually for further configuration and workflow details.

Our team put together these tutorials for (a) ourselves as scientists who want to learn about new topics in drug design and how to actually apply them practically to data using Python/KNIME, (b) new students in the group who need a compact but detailed enough introduction to get started with their project, and (c) for the classroom where we can use the material directly or build on top of it.

The pipeline is illustrated using the epidermal growth factor receptor (EGFR), but can easily be applied to other targets of interest. Topics include how to fetch, filter and analyze compound data associated with a query target. The bundled project including all workflows is freely available on KNIME Hub. The Hub also lists the individual workflows for separate downloads if desired. Further details are given in the following sections.

**Note:** The screenshots shown below are taken from the individual workflows, which resemble the complete workflow but have different input and output sources.

## Workflow 1: Acquire compound data from ChEMBL

Information on compound structure, bioactivity, and associated targets are organized in databases such as ChEMBL, PubChem, or DrugBank. Workflow W1 shows how to obtain and preprocess compound data for a query target (default target: EGFR) from the ChEMBL web services.

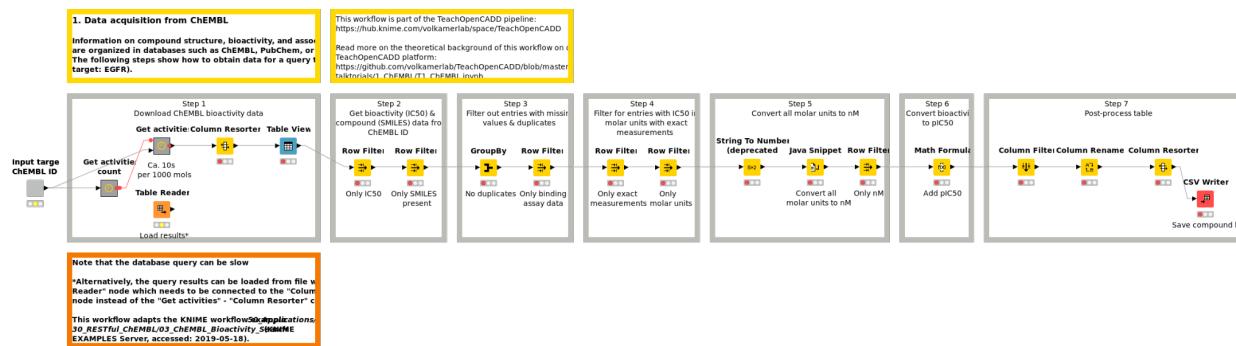


Figure 2 Workflow 1. Acquire compound data from ChEMBL

## Workflow 2: Filter datasets by ADME criteria

Not all compounds are suitable starting points for drug development due to undesirable pharmacokinetic properties, which for instance negatively affect a drug's absorption, distribution, metabolism, and excretion (ADME). Therefore, such compounds are often excluded from data sets for virtual screening. Workflow W2 shows how to remove less drug-like molecules from a data set using Lipinski's rule of five.

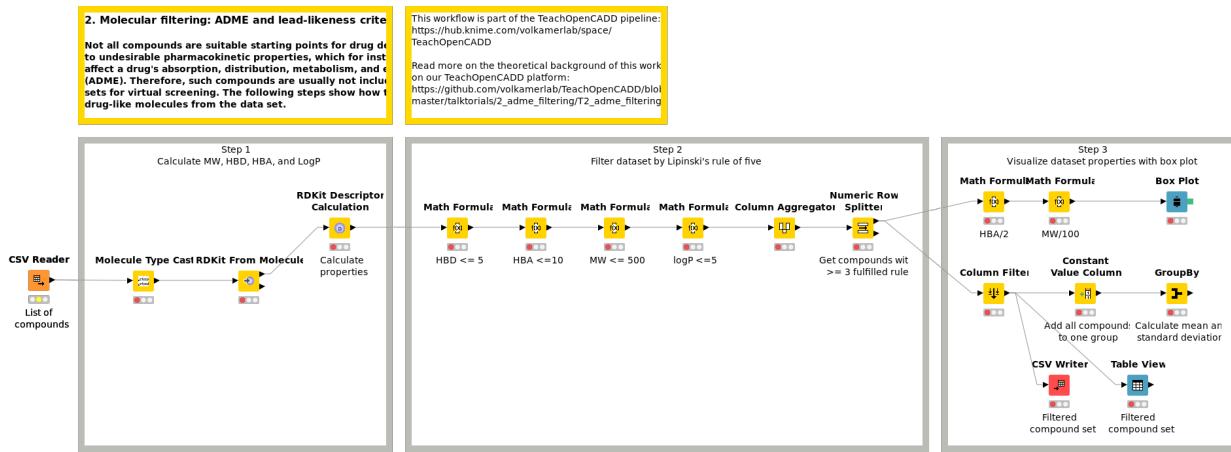


Figure 3 Workflow 2 Filter data sets by ADME criteria

## Workflow 3: Set alerts based on unwanted substructures

Compounds can contain unwanted substructures that may cause mutagenic, reactive, or other unfavorable pharmacokinetic effects or that may lead to non-specific interactions with assays (PAINS). Knowledge on unwanted substructures in a data set can be integrated in cheminformatics pipelines to either perform an additional filtering step before screening or - more often - to set alert flags to compounds being potentially problematic (for manual inspection by medicinal chemists). Workflow W3 shows how to detect and flag such unwanted substructures in a compound collection.

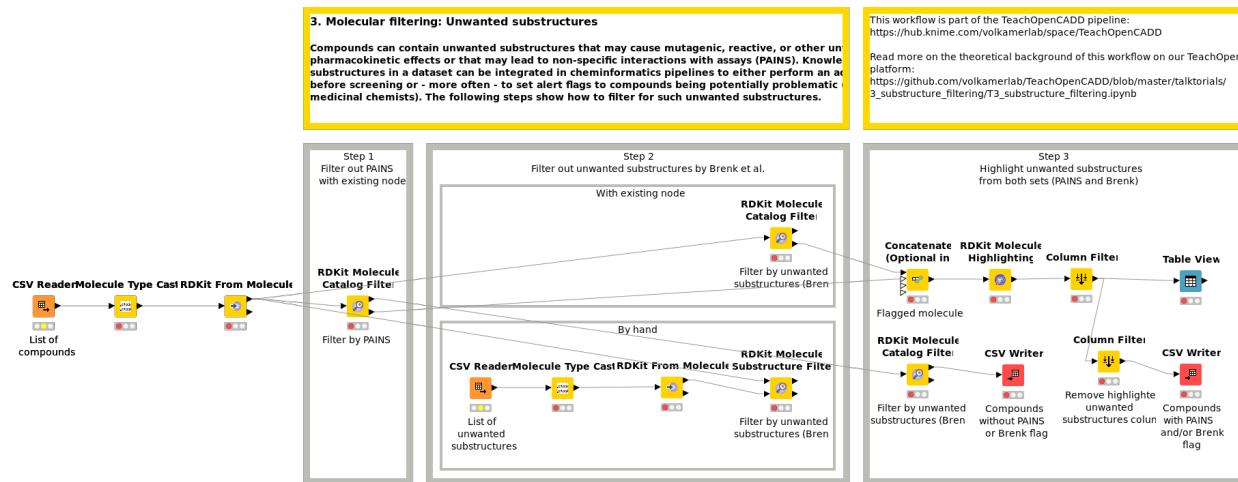


Figure 4 Workflow 3 Set alerts based on unwanted substructures

## Workflow 4: Screen compounds by compound similarity

In virtual screening (VS), compounds similar to known ligands of a target under investigation often build the starting point for drug development. This approach follows the similar property principle stating that structurally similar compounds are more likely to exhibit similar biological activities. For computational representation and processing, compound properties can be encoded in the form of bit arrays, so-called molecular fingerprints, e.g. MACCS and Morgan fingerprints. Compound similarity can be assessed by measures such as the Tanimoto and Dice similarity. Workflow W4 shows how to use these encodings and comparison measures. VS is here conducted based on a similarity search.

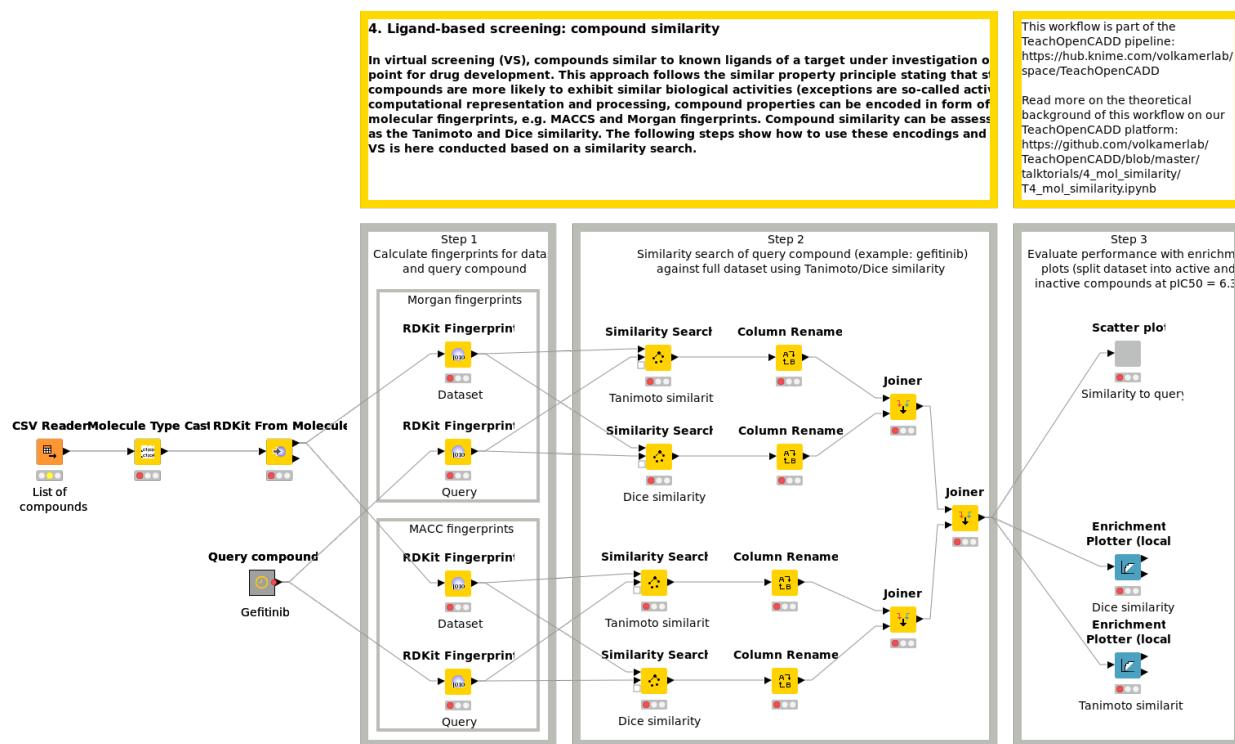


Figure 5 Workflow 4 Screen compounds by compound similarity

## Workflow 5: Group compounds by similarity

Clustering can be used to identify groups of similar compounds, in order to pick a set of diverse compounds from these clusters for e.g. non-redundant experimental testing or to identify common patterns in the data set. Workflow W5 shows how to perform such a clustering based on a hierarchical clustering algorithm.

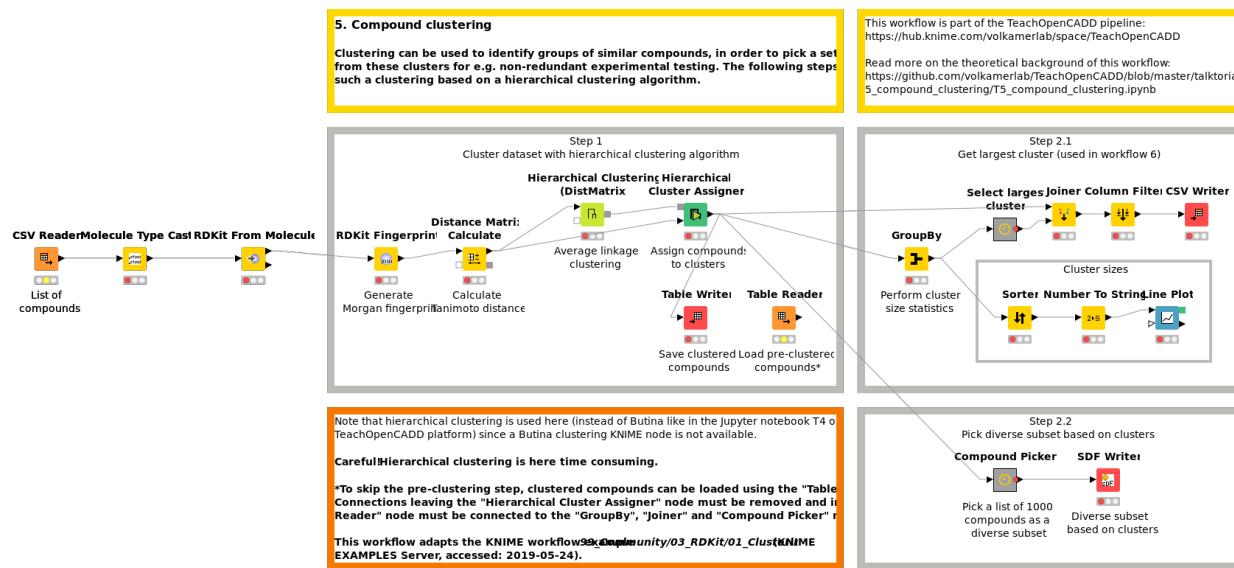


Figure 6 Workflow 5 Group compounds by similarity

# Workflow 6: Find the maximum common substructure in a collection of compounds

In order to visualize shared scaffolds and thereby emphasize the extent and type of chemical similarities in a compound cluster, the maximum common substructure (MCS) can be calculated and highlighted. In Workflow W6, the MCS for the largest cluster from previously clustered compounds (W5) is calculated using the FMCS algorithm.

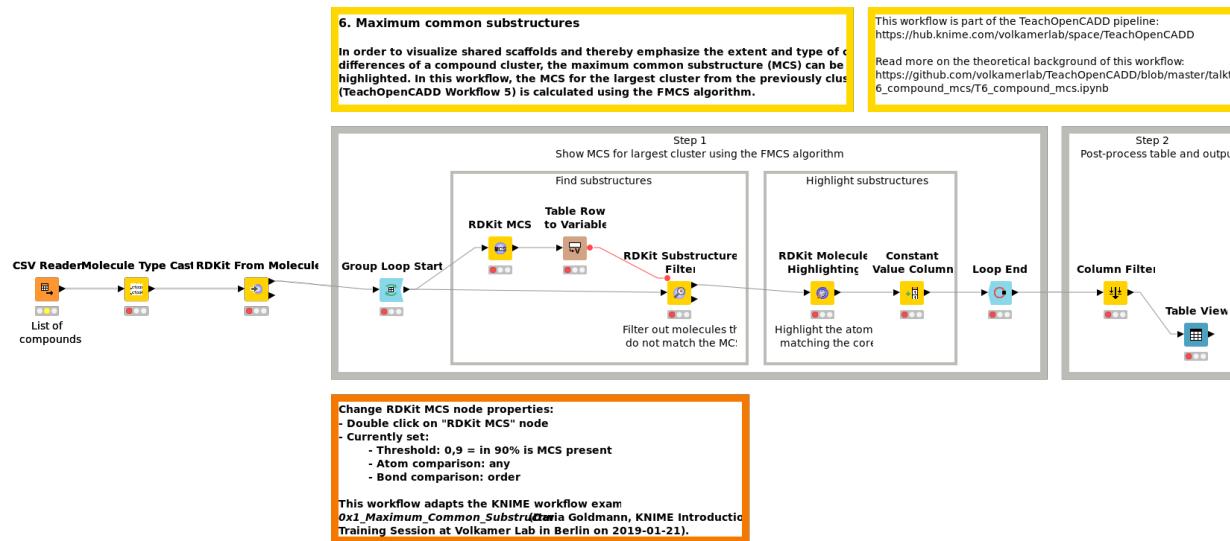


Figure 7 Workflow 6 Find the maximum common substructure in a collection of compounds

# Workflow 7: Screen compounds using machine learning methods

With the continuously increasing amount of available data, machine learning (ML) gained momentum in drug discovery and especially in ligand-based virtual screening (VS) to predict the activity of novel compounds against a target of interest. In Workflow W7, different ML models (RF, SVM and NN) are trained on the filtered ChEMBL dataset to discriminate between active and inactive compounds with respect to a protein target.

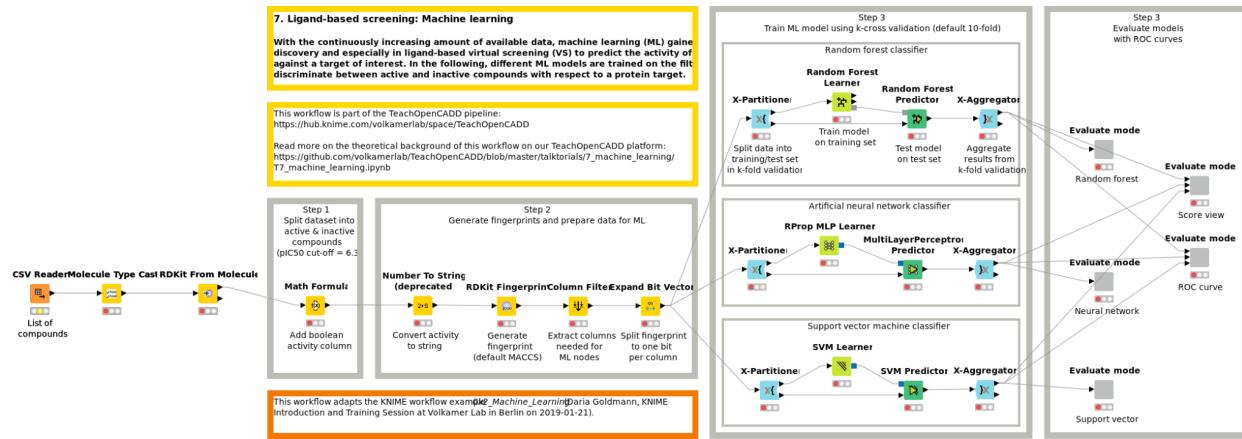


Figure 8 Workflow 7 Screen compounds using machine learning methods

## Workflow 8: Acquire structural data from PDB

The PDB database holds 3D structural data and meta information on experimentally resolved proteins. Workflow W8 shows how structural data can be automatically fetched from the PDB and processed.

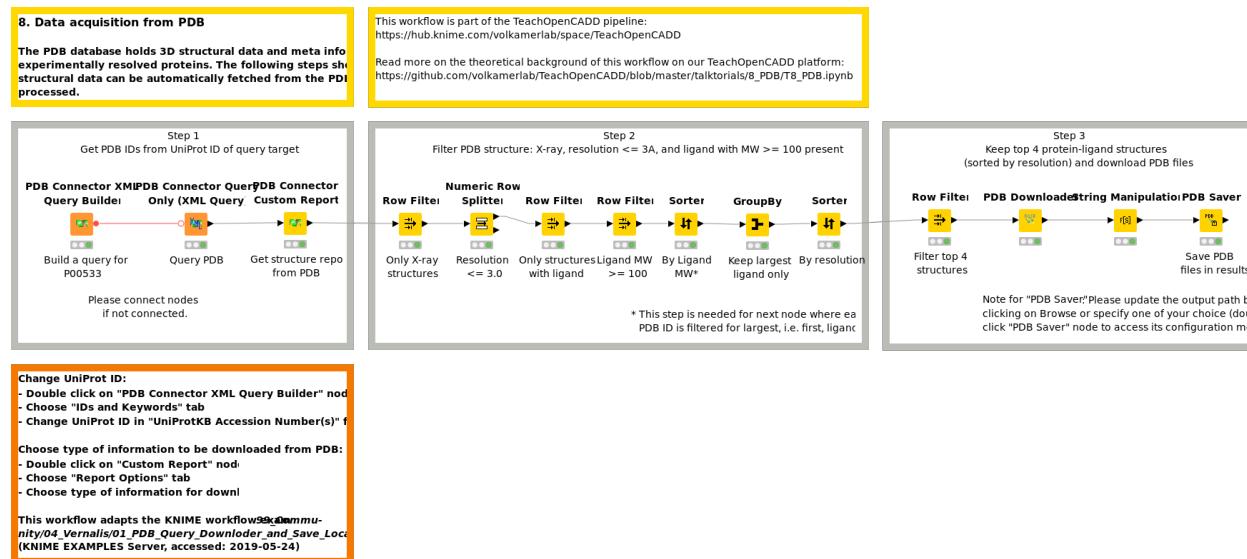


Figure 9 Workflow Acquire structural data from PDB

## Requirements

All the workflows have been updated for KNIME version 4.3. In addition to some extensions provided by the KNIME team, TeachOpenCADD also requires:

- RDKit KNIME integration, by NIBR
- Vernalis KNIME nodes, by Vernalis Research.

For a full list of requirements, please check our project on the KNIME Hub.

## References

Sydow, D., Morger, A., Driller, M., & Volkamer, A. (2019). TeachOpenCADD: a teaching platform for computer-aided drug design using open source packages and data. *Journal of cheminformatics*, 11(1), 1-7.

Sydow, D., Wichmann, M., Rodríguez-Guerra, J., Goldmann, D., Landrum, G., & Volkamer, A. (2019). TeachOpenCADD-KNIME: a teaching platform for computer-aided drug design using KNIME workflows. *Journal of chemical information and modeling*, 59(10), 4083-4086.

# 1.2 Training a machine learning model - to building a predictive web application in three steps

By Janina Mothes

Find the workflow(s) here: <https://kni.me/w/VxE7Y-PGz8jP6LaU>

In [earlier posts](#) on the KNIME blog we introduced a parameter optimization workflow that uses four different machine learning models, individually optimizes the hyperparameters, and picks the best combination of machine learning method and hyperparameters for the user. However, the usual data scientist journey doesn't end here. Having spent hours, days, or even (sleepless) nights exploring your data and finding the model with the best hyperparameters, you don't want your model to be buried and forgotten somewhere on your computer. In real world situations, you would want to make your model predictions accessible to other people. Maybe even create a web application, where users could enter their data and get your model prediction with some helpful data visualizations.

Here we will show you how to go from your trained model to a predictive web application with KNIME in three easy steps:

1. Model selection and parameter optimization
2. Integrated Deployment of the best performing model
3. Creating a web application

## 1. Model selection and parameter optimization

We took the [Model Optimization and Selection workflow](#) (shown in Figure 10) from a previous blog post as a starting point and did some adjustments.

Like the old workflow, the adjusted workflow starts by reading in the same data. The dataset is a subset of 844 compounds, i.e. molecules, from a public data set available [here](#) (data set 19), which were tested for activity against [CDPK1](#). Each compound is either "active" or "inactive".

In contrast to the original workflow, we now integrated the data preprocessing. In the "Generate Fingerprints" metanode, we compute five molecular fingerprints for the compounds using the RDKit nodes.

#### Model Selection with Integrated Deployment

This workflow deploys an advanced parameter optimization protocol with four machine learning methods. The model performances can be compared in the interactive view of the "Pick best Model" component. Based on the selection in the interactive view, the selected model is build and automatically deployed. Additionally, a model report is generated.

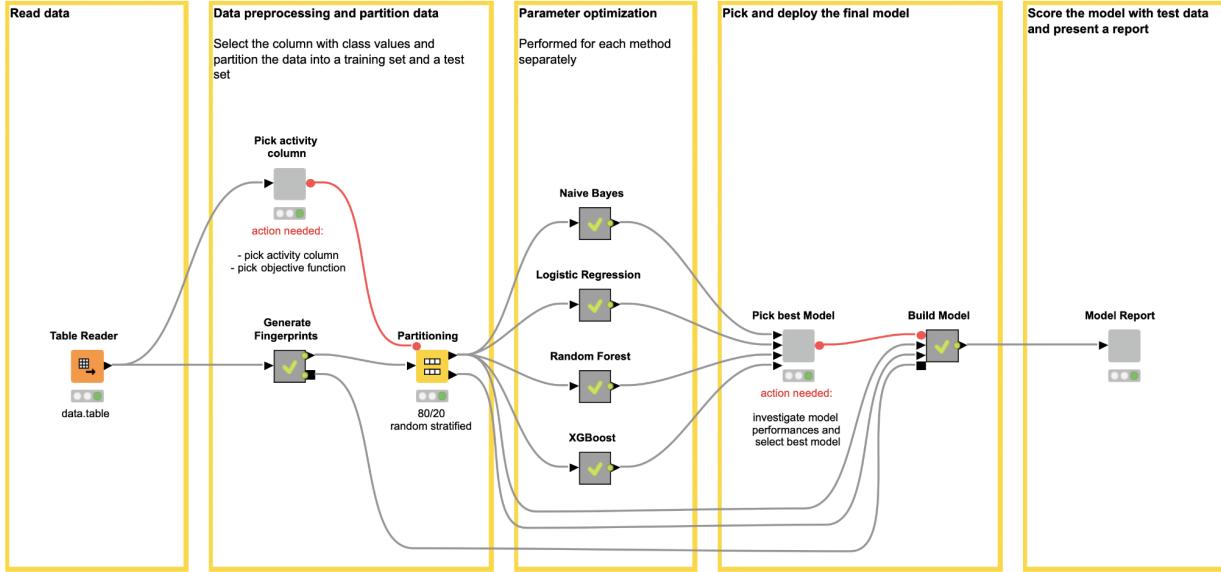


Figure 10 Overview of the adjusted Model Optimization and Selection workflow. Each machine learning method is encapsulated within corresponding metanodes

In the “Pick activity column” component the target column containing the class values (active, inactive) can be selected by the user (see Figure 11). Additionally, the user can select the objective function value. The objective function value is the value that will be optimized for each of the machine learning models. It is handy to define the objective function value beforehand in one place. This reduces the possibility of introducing mistakes by accidentally selecting different objective function values for each machine learning method. In our example, we choose an Enrichment Factor of 5% as the objective function value.

Based on our target column (i.e. activity), we perform a stratified partitioning of the data into a:

- training set for parameter optimization and a
- test set for scoring the best model.

## Hyperparameter optimization

Similar to the original workflow, the hyperparameter optimization for each of the four machine learning methods is done in the corresponding metanodes: Naive Bayes, Logistic Regression, Random Forest and XGBoost. Please have a look at the [previous blog post](#) for a detailed description of the parameter optimization cycles.

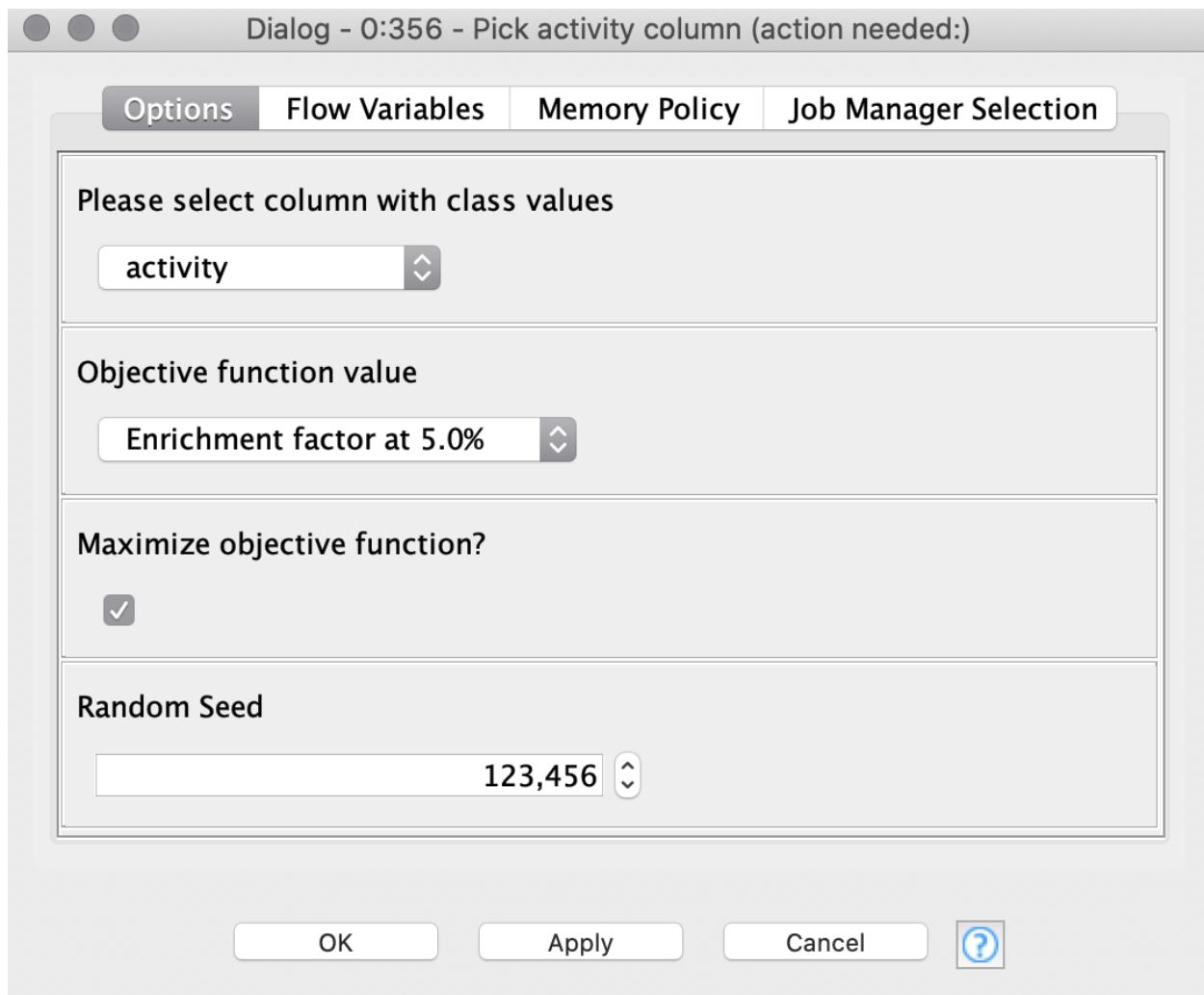


Figure 11 Configuration dialog of the “Pick activity column” component. Besides selecting the activity column, it also provides the option to select the objective function value.

## Compare Model Performances

After optimizing each machine learning method, we want to compare the model performances and select the best model for our use case.

- The “Pick best model” component provides an interactive view, which helps us evaluate the performance of each model (Figure 12).
- The Parallel Coordinates Plot shows the values for the different model performance measures.

In our example, we optimized the models for the enrichment factor at 5% and all models perform equally well regarding both the enrichment factor at 5% as well as the discovery rate at 5%. Therefore, we have to take additional model performance measures into account. For our example, we will mainly focus on:

- Cohen's kappa
- F-measure
- Balanced accuracy

since those model performance measures are especially suited for imbalanced classes (as is the case for our dataset). For Cohen's kappa and F-measure, the XGBoost model outperforms the other models. In terms of balanced accuracy, the XGBoost is the second best model. Therefore, we select the XGBoost model as the best model for our use case.

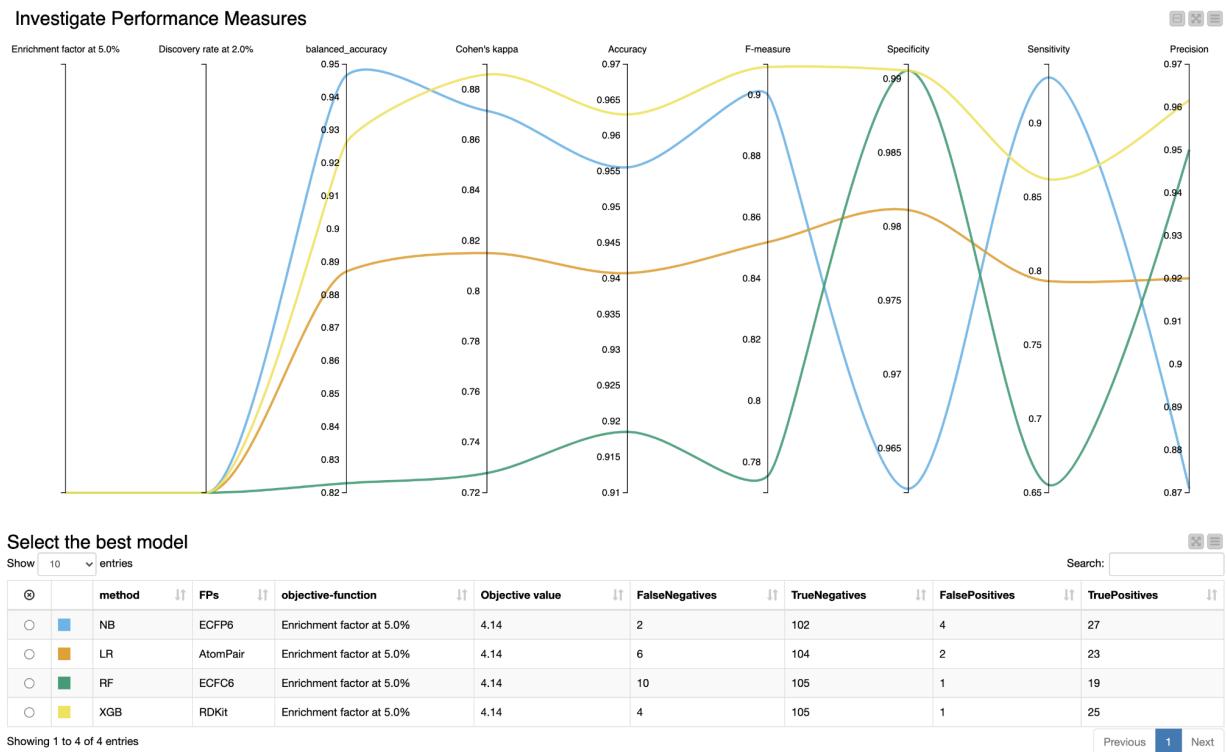


Figure 12 Interactive view of the “Pick best model” component showing the performances of all models in the Parallel Coordinate Plot. The objective function value (e.g. Enrichment Factor at 5%), is equally optimal in all models. The best model can be selected in the Table View.

## Report on Best Hyperparameter and Model Performance

The last component, “Model Report”, creates a short report containing information about the best hyperparameter and performance of the model (see Figure 13). In our example, the XGBoost model using RDKit fingerprint was selected as it outperformed the other models in regards to Cohen's Kappa.

# Model Report for Assay: CDPK1

Confusion Matrix

| Real     | Predicted |          |      |
|----------|-----------|----------|------|
|          | active    | inactive |      |
| active   | 33        | 3        | 0.92 |
| inactive | 9         | 124      | 0.93 |
|          | 0.79      | 0.98     |      |

## Performance:

**AUC:** 0.976  
**Enrichment factor at 1%, 5%, 10%:** 2.778, 4.444, 3.889  
**Balanced Accuracy:** 0.924  
**Cohen's kappa:** 0.924  
**F-Score:** 0.846

## Assay Details

**Description:** CDPK1  
**actives:** 36; **inactives:** 133  
**objective:** Enrichment factor at 5.0%

## Model Details

**Fingerprints:** RDKit  
**Method:** XGBoost  
**maxDepth:** 3; **learning\_rate:** 0.05 **boosting:** 50

Figure 13 The model report view showing performance of the final model.

## 2. Integrated Deployment of the best performing model

To deploy your trained model, there are a few things that you need to keep in mind. The deployed workflow needs to include not only your trained model but also any data preprocessing that you did before training the model. Before KNIME Analytics Platform Version 4.2.0, that meant saving your trained model to a file and copying your preprocessing nodes to the deployed workflow, which increased the risk of mistakes.

Starting with KNIME Analytics Platform 4.2.0, there are some new nodes which can make your life much easier and help you to deploy your model in a more automated way ([KNIME Integrated Deployment](#)). Now you can use the Capture Workflow Start and End nodes to define the workflow parts that you want to deploy. This means that the workflow parts that you captured will be written automatically to a separate workflow.

In our example, we capture the data preprocessing (e.g. computation of the five fingerprints) with the Capture Workflow Start and End nodes by simply inserting them at the beginning and end of our data preprocessing workflow, as shown in Figure 14. Similarly, model prediction is captured

using the same nodes (Figure 15). The Workflow Combiner node is used to join the two workflows. The red Workflow Writer node deploys the workflow automatically to the user-defined destination. That's all you need to do.

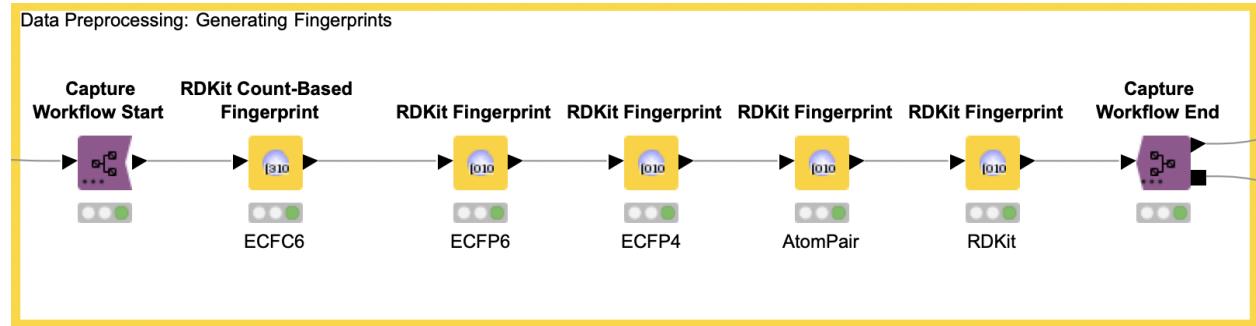


Figure 14 Capturing the data preprocessing (computing the five fingerprints) with the Capture Workflow Start node and the Capture Workflow End node

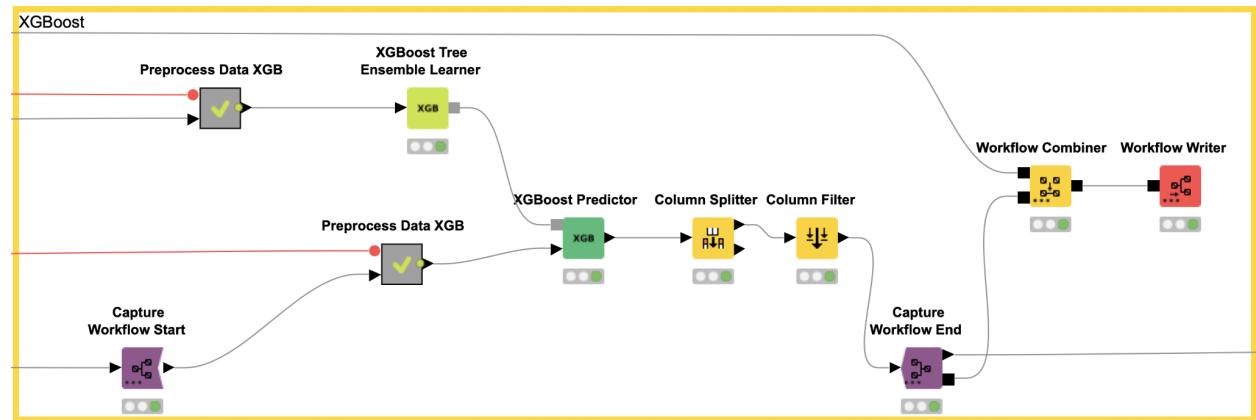


Figure 15 Capturing the model prediction with the purple Capture Workflow Start node and the purple Capture Workflow End node. Combining the data preprocessing part and the model prediction part using the yellow Workflow Combiner node. The red Workflow Writer node deploys the combined workflow

The deployed workflow is shown in Figure 16. The Container Input (Table) and Container Output (Table) nodes are added automatically. Those nodes enable the deployed workflow to receive a table from an external caller via a REST API and return the model prediction. At this point, we could deploy our model as a REST web service by simply copying the deployed workflow to a KNIME Server. This way, anyone could get predictions from our trained model using the [KNIME Server REST API](#). However, most people do not use REST APIs but rather want a graphical interface where they can easily enter their data and get model predictions together with a useful [visualization](#). So, let's move on to the last step and create a small web application.

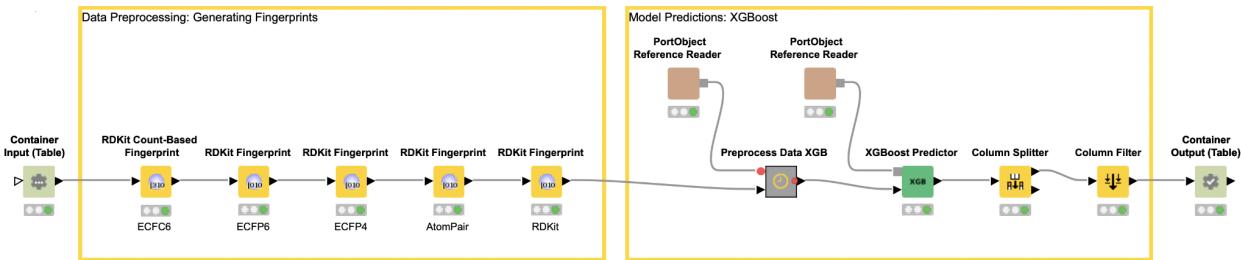


Figure 16 Deployed workflow combining the data preprocessing (computation of the five fingerprints) and the model prediction.

### 3. Creating a web application

To create a web application you need the [KNIME Webportal](#) which comes along with the commercial KNIME Server license. In case you are working with the open source KNIME Analytics Platform, you can still create and execute the web application workflow (see Figure 17) but you won't be able to deploy the actual web application.

The workflow (shown in Figure 17) creates three views/user interactions points for our web application. The views are generated by components. The first component/view “File Input” (shown in Figure 18) enables the user to upload a file containing the compounds for which they want a model prediction (CDPK1 activity prediction).

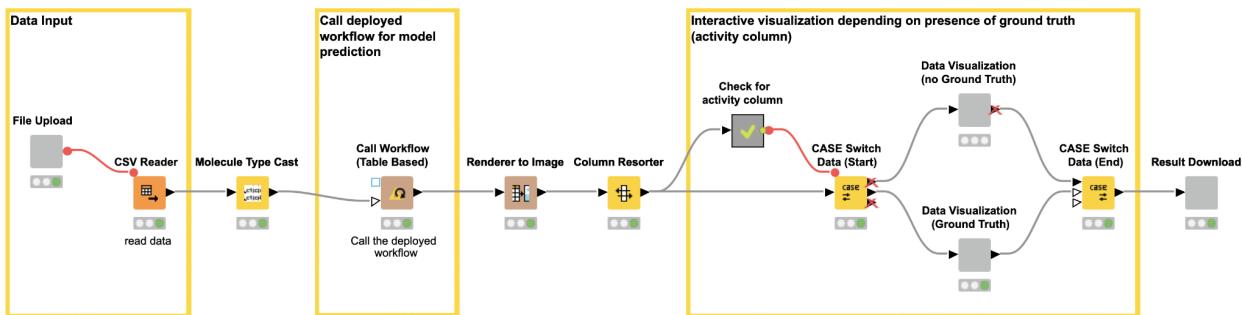


Figure 17 Web application workflow for the KNIME Webportal.

After the file upload, there are some nodes that process the input in the background and are not visible in the web application. For example, the Molecule Type Cast node converts the SMILES from a string format into a SMILES data format. The Call Workflow (Table Based) node passes the compounds to our deployed workflow (Figure 16). There the data is preprocessed (computing the molecular fingerprints) and the CDPK1 activity is predicted using the best performing model, in our case XGBoost. The predictions for each compound are returned as an additional column to our input table.

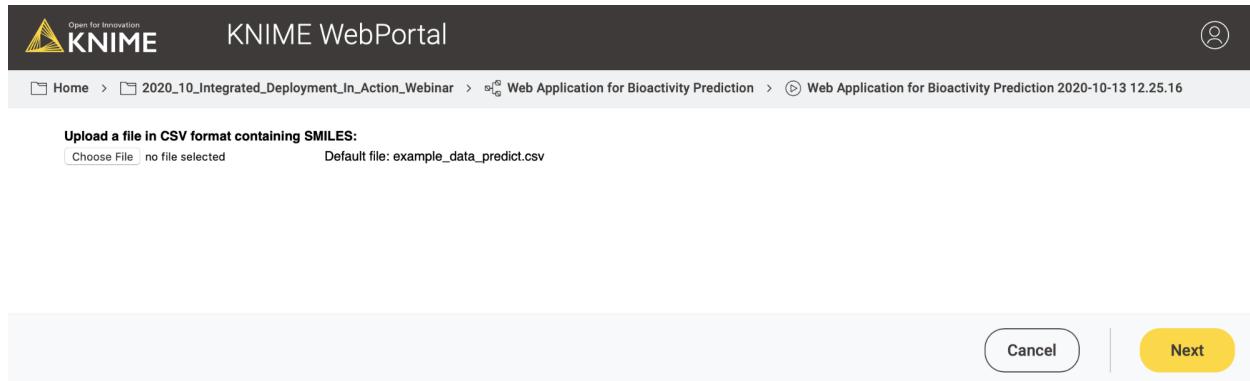


Figure 18 First view of the web application. The user can select the file containing the data for the model prediction

For the second view of the web application, we use a Case Switch node depending on the data that is uploaded by the user.

In the first case, the user uploads a data file which contains simply chemical structures, without any knowledge about its activity on CDPK1. The user simply wants a prediction for his/her data.

In the second case, the user might want to test or evaluate our model (some people like to do that). In that case, they will upload a file which, besides the chemical structures, also contains the true activity ("ground truth") on CDPK1. If the activity column ("ground truth") is available, we can also provide the confusion matrix and the ROC curve for the data (see Figure 20). So our workflow contains a metanode in which we determine if the activity column is available in the uploaded data file. Depending on the presence of the activity column, a different web application view is shown (Figure 19 vs Figure 20).

In the third and last view of the web application, the user can simply download the results of the model predictions including his/her selection in the interactive view as an Excel file (Figure 21).

## Wrapping up

We showed you three simple steps to automatically get from training your machine learning model to building a predictive web application. You can adjust your parameter optimization and model selection pipeline by using the new integrated deployment nodes to automatically deploy the best model for your data. If you create a simple web application workflow for the KNIME Webportal you can easily make your trained model accessible to “consumers”.

The screenshot shows the KNIME WebPortal interface. At the top, there is a navigation bar with the KNIME logo and a user icon. Below the navigation bar, the URL is visible: Home > 2020\_10\_Integrated\_Deployment\_In\_Action\_Webinar > Web Application for Bioactivity Prediction > Web Application for Bioactivity Prediction 2020-10-13 12.21.54. The main content area contains two tables. The first table has columns for a checkbox, an orange square icon, and the text "Prediction (activity)" with rows for "active" (count 20) and "inactive" (count 64). The second table has columns for a checkbox, an orange square icon, "Image", "compound\_chembl\_id", "TCMDC\_ID", "Prediction (activity)", "P (activity=inactive)", and "P (activity=active)". It lists two compounds: CHEMBL588540 and CHEMBL581801. For each compound, it shows its SMILES string, TCMDC ID, prediction (active), and probabilities for inactive and active states. At the bottom of the page are "Back", "Cancel", and "Next" buttons.

|                                     |   | Prediction (activity) |  | count |
|-------------------------------------|---|-----------------------|--|-------|
| <input checked="" type="checkbox"/> | <span style="background-color: orange; width: 10px; height: 10px; display: inline-block;"></span> | active                |  | 20    |
| <input type="checkbox"/>            | <span style="background-color: blue; width: 10px; height: 10px; display: inline-block;"></span>   | inactive              |  | 64    |

|                                     |   | Image   | compound_chembl_id | TCMDC_ID     | Prediction (activity) | P (activity=inactive) | P (activity=active) |
|-------------------------------------|---|---|--------------------|--------------|-----------------------|-----------------------|---------------------|
| <input checked="" type="checkbox"/> | <span style="background-color: orange; width: 10px; height: 10px; display: inline-block;"></span> |  | CHEMBL588540       | TCMDC-133559 | active                | 0.4556081295013428    | 0.5443918704986572  |
| <input checked="" type="checkbox"/> | <span style="background-color: orange; width: 10px; height: 10px; display: inline-block;"></span> |  | CHEMBL581801       | TCMDC-134114 | active                | 0.18771350383758545   | 0.8122864961624146  |

Figure 19 Option 1 for the second view of the web application. If the activity column is not included, the user can interactively select the compounds of interest

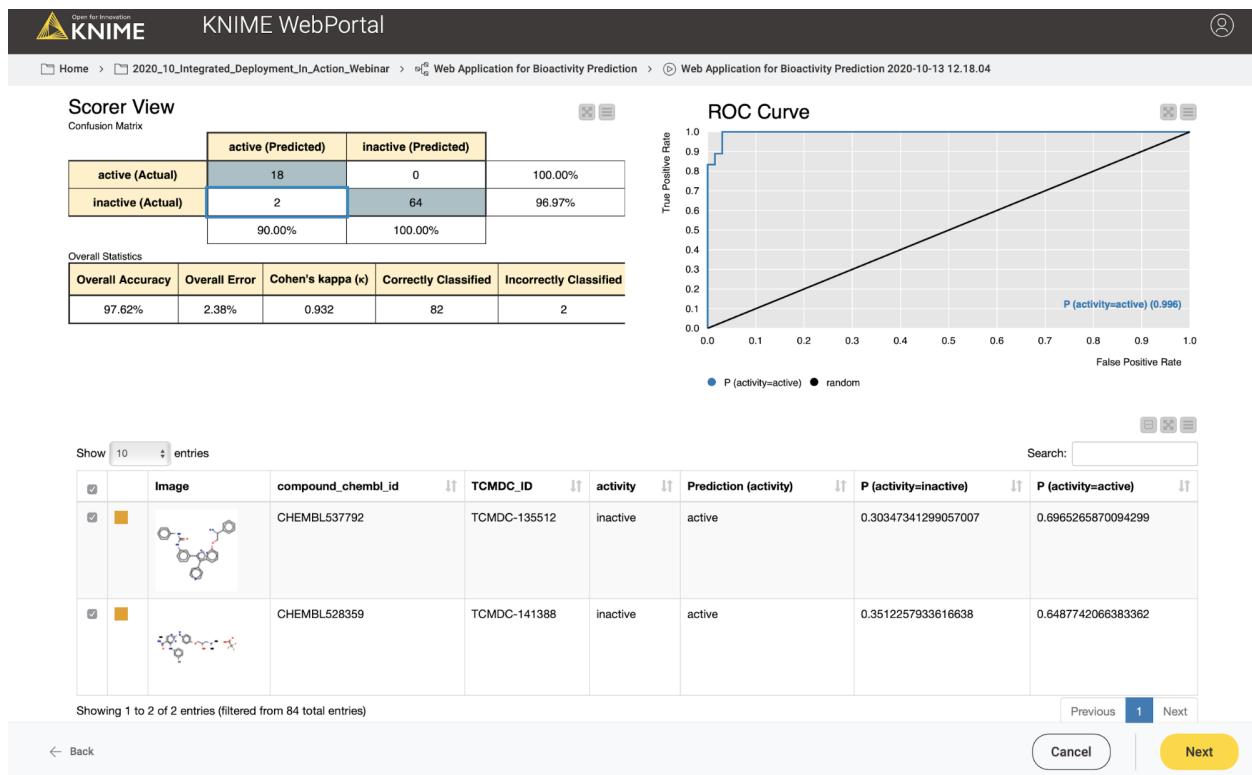


Figure 20 Option 2 for the second view of the web application. If the activity column is included in the uploaded data set, the user can interactively explore the model predictions.

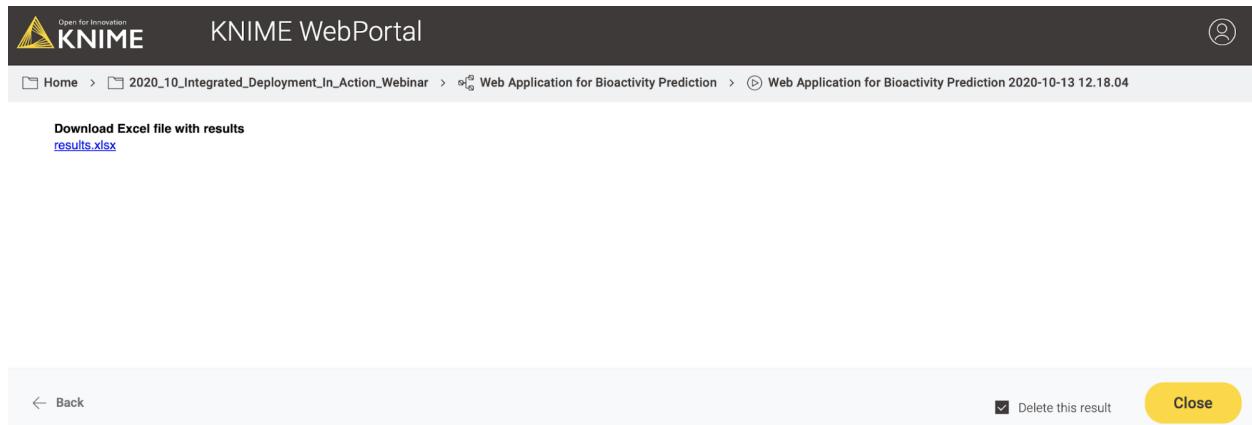


Figure 21 Last view of the web application: User can download the results of the model prediction as an Excel file

# 1.3 Multitasking doesn't always make things worse: interactive bioactivity prediction with multitask neural networks

By Greg Landrum

Find the workflow(s) here: <https://kni.me/w/ocNbAaAkXf1aOGBo>

A [CHEMBL-OG post](#) 'Multi-task neural network on ChEMBL with PyTorch 1.0 and RDKit' by Eloy, from way back in 2019 showed how to use data from ChEMBL to train a multi-task neural network for bioactivity prediction - specifically to predict targets where a given molecule might be bioactive. Eloy has links to more info in his blog post, but multi-task neural nets are quite interesting because the way information is transferred between the different tasks during training can result in predictions for the individual tasks that are more accurate than what you'd get if you just built a model for that task alone.

It's a big contrast to most humans: our performance tends to go down the moment we start multitasking. In any case, I find this an interesting problem and Eloy provided all the code necessary to grab the data from ChEMBL and reproduce his work, so I decided to pick this up and build a KNIME workflow to use the multitask model. For once I didn't have to spend a bunch of time with data prep (thanks Eloy!) so I could directly use Eloy's Jupyter notebooks to train and validate a model. After letting my workstation churn away for a while I had a trained model ready to go; now I just needed to build a prediction workflow.

## Loading the network and generating predictions

Eloy's notebooks build the multitask neural network using PyTorch, which KNIME doesn't directly support, but fortunately both KNIME and PyTorch support the ONNX ([Open Neural Network Exchange](#)) format for interchanging trained networks between neural network toolkits. So I was able to export my trained PyTorch model for bioactivity prediction into ONNX, read that into KNIME with the ONNX Network Reader node, convert it to a TensorFlow network with the ONNX to TensorFlow Network Converter node, and generate predictions using the TensorFlow Network Executor node.

Now that I have the trained network loaded into KNIME, I need to create the correct input for it. Since the model was trained using the RDKit this is quite easy using the RDKit KNIME Integration.

I know that the model was trained using the RDKit's Morgan fingerprint with a radius of 2 and a length of 1024 bits and I can generate the same fingerprints with the RDKit Fingerprint node. Since I can't pass fingerprints directly to the neural network, I also add an Expand Bit Vector node to convert the individual bits in the fingerprints into columns in the input table. The compounds that we'll generate fingerprints for are read in from a text file containing SMILES and a column with

compound IDs that we'll use as names. The sample dataset used in this blog post (and for the example workflow) is made up of a set of molecules exported from ChEMBL and a couple of invented compounds I created by manually editing ChEMBL molecules.

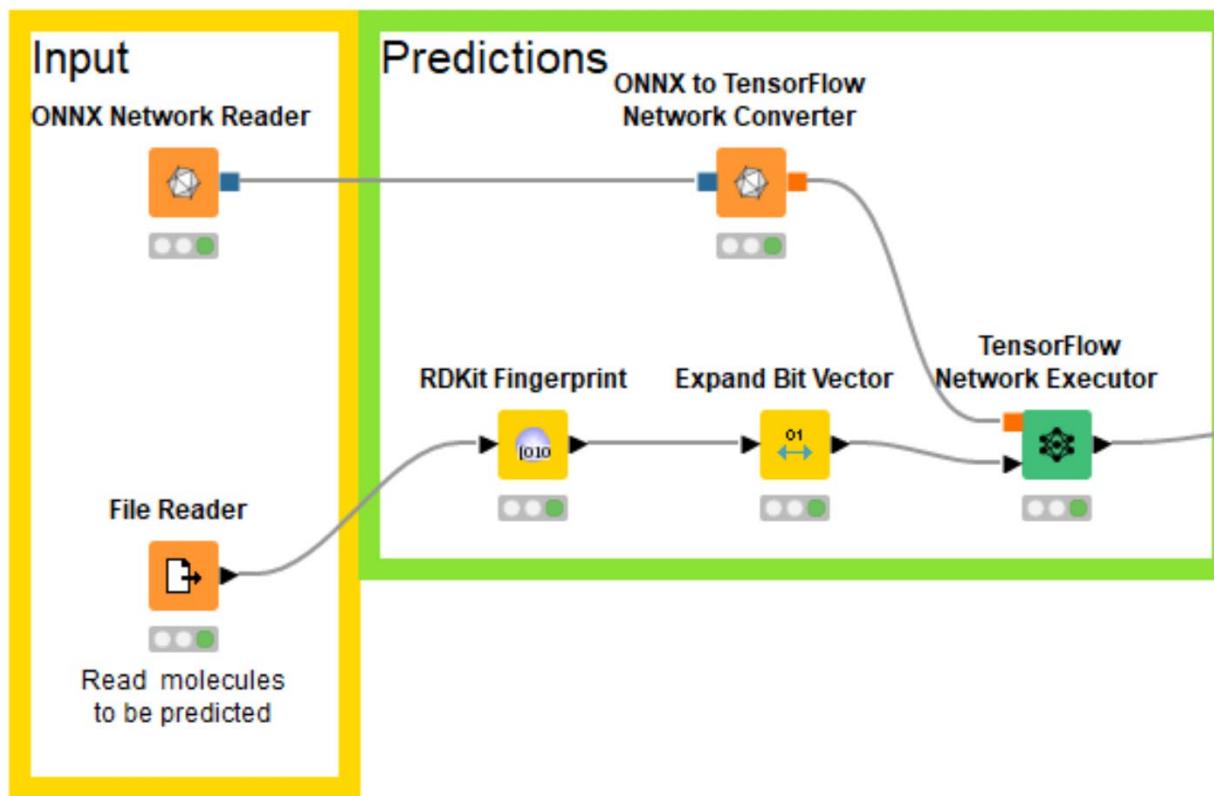


Figure 22 Here we see the part of the workflow that handles both loading the neural network and preparing the input for it.

The output of the TensorFlow Network Executor node is a table with one row for each molecule we generated a prediction for and one column for each of the 560 targets the model was trained on. The cells contain the scores for the compounds against the corresponding targets (Figure 23).

| Table "default" - Rows: 12 Spec - Columns: 560 Properties Flow Variables |          |          |          |          |          |
|--|----------|----------|----------|----------|----------|
| Row ID   | D 1130_0 | D 1132_0 | D 1134_0 | D 1136_0 | D 1138_0 |
| Row0   | 0.369    | 0.096    | 0.026    | 0.086    | 0.583    |
| Row1   | 0.015    | 0.022    | 0.809    | 0.047    | 0.027    |
| Row2   | 0.008    | 0.024    | 0.883    | 0.021    | 0.105    |
| Row3   | 0.003    | 0.078    | 0.192    | 0.061    | 0.014    |
| Row4   | 0.093    | 0.018    | 0.174    | 0.181    | 0.108    |

Figure 23 Predictions from the TensorFlow Network Executor node.

At this point we have a pretty minimal prediction workflow: we can use the multitask neural network to generate scores for new compounds. In the rest of this post I'll show a couple of ways to present the results so that it's a bit easier for people to interactively work with them.

## Showing the predictions in an interactive heatmap

The first interactive view that we'll use to display the predictions from the multitask neural network includes a heatmap with the predictions themselves and a tile view showing the molecules the predictions were generated for. The heatmap has the compounds in rows and targets in columns with the coloring of the cell determined by the computed scores. The tile view is configured to only show the selected rows.

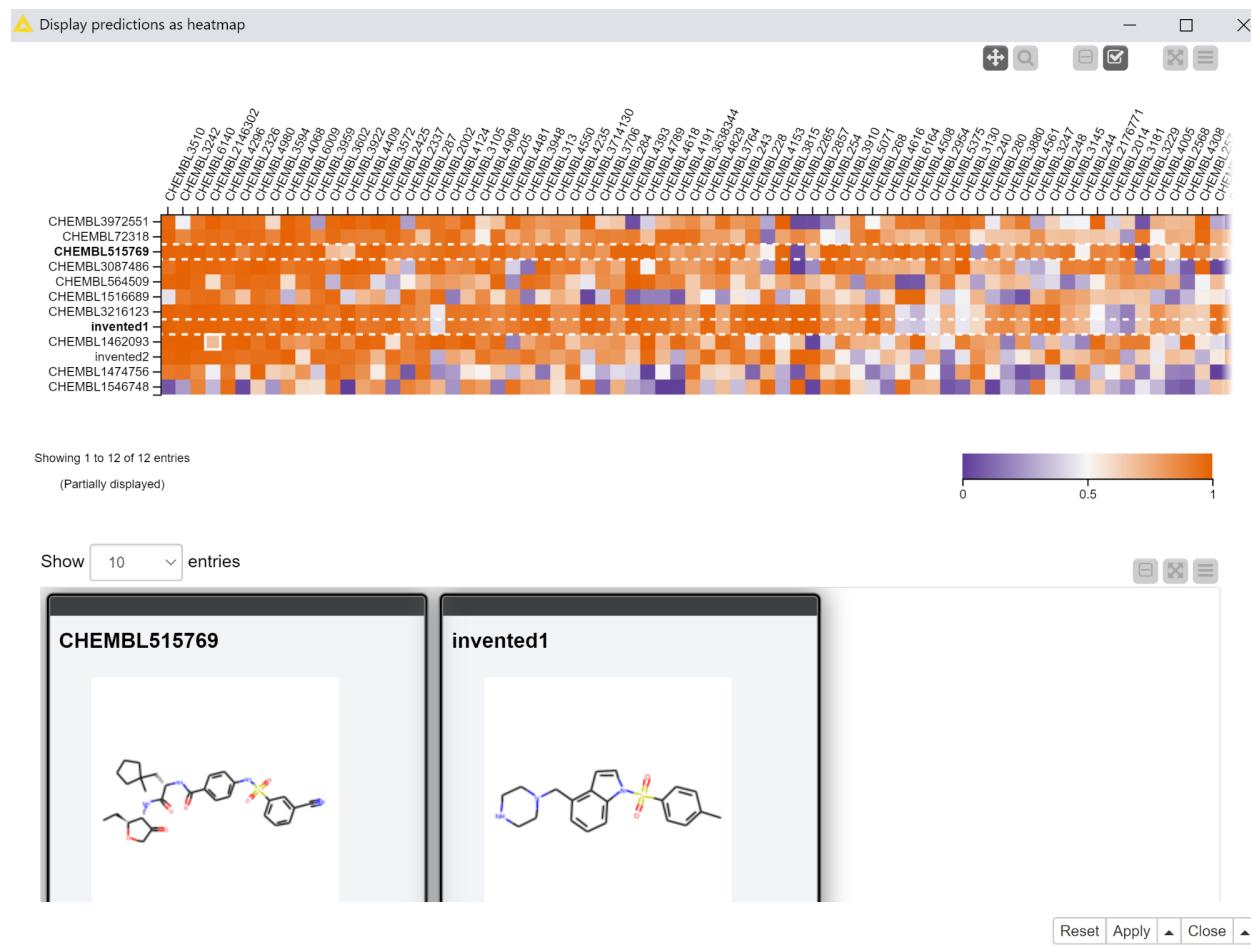


Figure 24 : Interactive view showing model predictions and the compounds.

The “Display predictions as heatmap” component that exposes this interactive view is set up so that only selected rows are passed to its output port. So in the example shown in Figure 24, there would only be two rows in the output of the “Display predictions as heatmap” component.

The workflow does a significant amount of data processing in order to construct the heatmap. I won't go into the details here, but the main work occurs in the "Reformat with bisorting" metanode, which reorders the compounds and targets based on their median scores. This brings targets which have more high-scoring compounds to the left of the heatmap and compounds with high scores against more targets to the top of the heatmap. Qualitatively the heatmap should get more red as you pan up and to the left and more blue as you pan down and to the right. There's no best answer as to the best sorting criteria for this purpose, so feel free to play around with the settings of the sorting nodes in the "Reformat with bisorting" metanode if you'd like to try something other than the median.

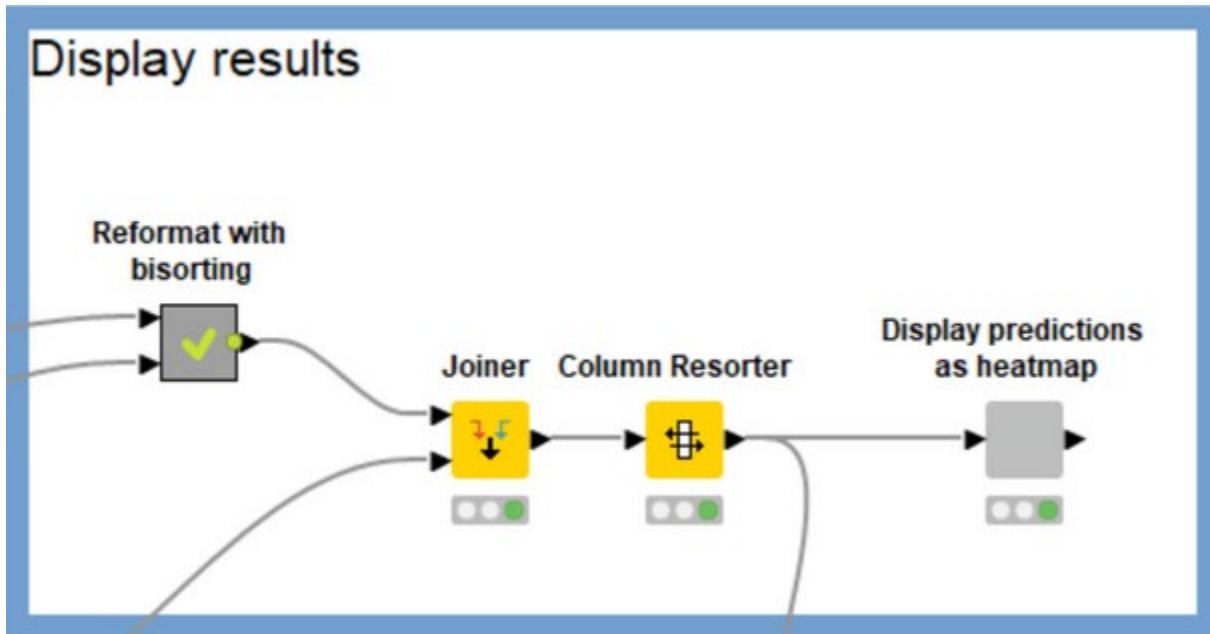


Figure 25 The part of the prediction workflow which generates the data for and displays the interactive heatmap view.

## Comparing predictions to measured values

A great way to gain confidence in a model's predictions is to compare them with measured data. Generally we can't do this, but sometimes there will be relevant measured data available for the compounds we're generating predictions for. In these cases it would be great to display that measured data together with the predictions. The remainder of the workflow is there to allow us to do just that (Figure 26).

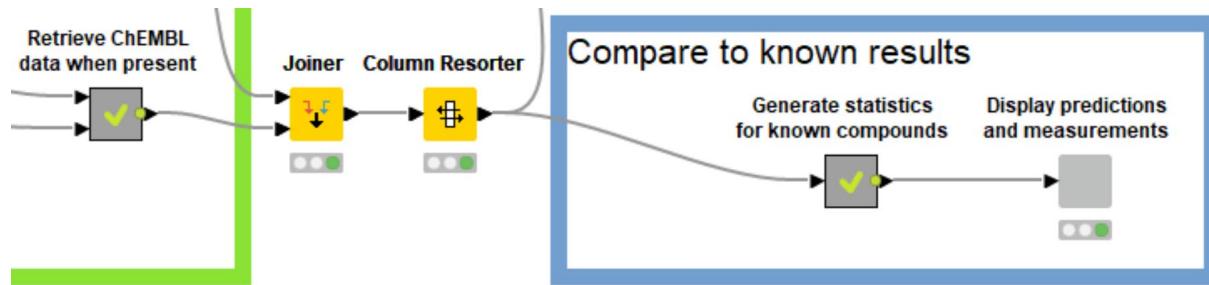


Figure 26 The part of the workflow for comparing predictions to measured data from ChEMBL.

This starts by generating InChI keys for the molecules in the prediction set, looking those up using the ChEMBL REST API, and then using the API again to find relevant activity data that was measured for those compounds. This is done in the "Retrieve ChEMBL data when present" metanode.

The output table of the metanode has one row for each compound, a ChEMBL ID for each compound that was found in ChEMBL, and one column for each target where there was experimental value in ChEMBL for one of the compounds in the prediction set. This data can be visualized, together with the predictions using the “Display predictions and measurements” component (Figure 27).

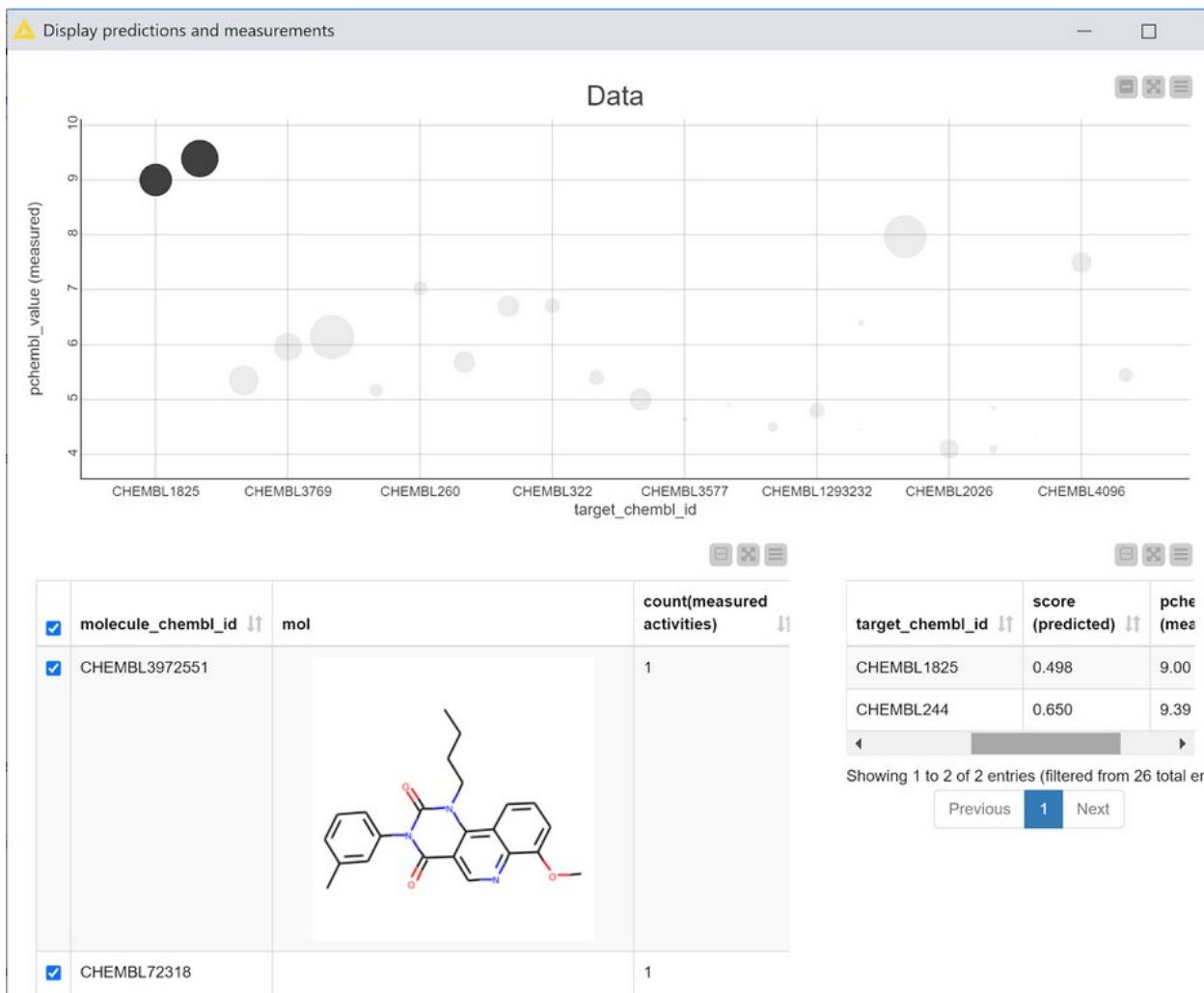


Figure 27 The “Display predictions and measurements” interactive view.

This interactive view is primarily based on the scatter plot at the top. Each point in the plot corresponds to one compound with data measured against one target. The ChEMBL IDs of the targets are on the X axis and the measured PCHEMBL values (as provided by the ChEMBL web services) are on the Y axis. The size of the points in the plot is determined by the calculated score of that compound for that target. The scatter plot is interactive: selecting points shows the relevant compounds in the table at the bottom left of the view and the corresponding scores and measured data in the table at the bottom right.

If the model is performing really well I'd expect the scatter plot in Figure 27 to have large scores (big points) for compounds which have high activity (large pChEMBL values), i.e. bigger points towards the top of the plot and smaller points towards the bottom. That's more or less what we observe. There are clearly some outliers, but it's probably still ok to pay at least some attention to the model's predictions for the other compounds/targets. (Note: This isn't a completely valid evaluation since most of the data points I'm using in this example were actually in the training set for the model. The example is shown here in order to demonstrate the view and its interactivity.)

## Wrapping Up

In this blog post I've demonstrated how to import a multitask neural network for bioactivity prediction built with PyTorch into a KNIME workflow and then use that to generate predictions for new compounds. I also showed a couple of interactive views for working with and gaining confidence in the model's predictions. The workflow, trained model, and sample data are available on the KNIME hub for you to download, learn from, and use in your own work.

## Chapter 2: Bioinformatics

In this chapter we want to highlight three interesting stories that will give insight into bioinformatics use cases with KNIME.

### Analyzing Gene Expression Data with KNIME

In the first story of this chapter we want to take a look at a very common bioinformatics task: analyzing RNA-Seq data to find differentially expressed genes between two conditions, in this case between tumors and matched normal tissue. The first step is the upload of csv files. Next, we find differentially expressed genes using the edgeR library in the R snippet node. The user can investigate those genes and select genes of interest based on statistics from the gene expression analysis. We then cluster the genes based on similar expression profiles and investigate their biological pathways. Finally, we search for compounds targeting the selected gene products using Google's BigQuery. You have the option to deploy the workflow to the WebPortal so that the domain expert can easily interact with the data and results through a web page.

### Gut Microbiome Analysis with KNIME Analytics Platform

In this story, we have a look at a more advanced/specific bioinformatics use case called taxonomic profiling. We compare the composition of gut microbiomes from 10 patients at different time points while they undergo fecal transplantation, based on the microbe's 16S-rRNA gene sequences. The story showcases how to get FASTQ sequences from the European Nucleotide Archive (ENA) via REST and FTP services. We then preprocess the data using the R package DADA2 within KNIME via the R scripting nodes to create an Amplicon Sequence Variant (ASV) table. To create a taxonomic profile by grouping the sequence count and deriving the percentage of each group of the chosen taxonomic rank. In the last step, we investigate the shift in gut microbiome composition with the help of multiple interactive visualizations. Also here you have the option to deploy the workflow to the WebPortal so that the domain expert can easily interact with the data and results through a web page.

### Variant Prioritization - Reproducible Workflow with Domain Expert Interaction

This story illustrates how to do a typical bioinformatics application in KNIME, variant prioritization. The input data is a VCF file. We filter variants using the command line tools Bcftools and VCFtools, and we display statistics about the data in an interactive view. We then predict variant effects through a shared component using Ensembl's Variant Effect Predictor (VEP) via their REST API. We create an interactive view through which the domain expert can then filter and manually annotate variants. With access to a KNIME Server, these interaction points can be made available on the KNIME WebPortal. Lastly, final results are summarized including the intermediate files, the final result file, and all commands used throughout the workflow.

## 2.1 Analyzing Gene Expression Data with KNIME

By Jeany Prinz

Find the workflow(s) here: <https://kni.me/w/vUgr-iyGudXur-1B>

### Express Yourself!

All individuals are unique and so are our data needs. From simple csv files to REST APIs to Google's BigQuery or using customized shared components, KNIME Analytics Platform offers many ways to access and analyze your data. Today, we will demonstrate how to access all of these aforementioned data sources through the use case of analyzing and annotating gene expression data.

#### Gene Expression Analysis

Gene expression analysis is widely used in bioinformatics because it enables researchers to find gene products with increased or decreased synthesis in individuals with e.g. particular diseases. Gene expression is the process by which information from a gene is used in the synthesis of a functional gene product.

As we learned in a previous blog article, [Motifs and Mutations - The Logic of Sequence Logos](#), DNA mutations yield different effects of varying impact. Some have no noticeable effect at all and some can lead to severe diseases. As we also saw in that blog post, mutations that change gene expression are often associated with harmful effects. Hence, analyzing gene expression data directly is a straightforward way to find connections between genes and diseases.

#### Transcription

The first step in gene expression is called transcription, during which DNA is transcribed to RNA. Advances in massively parallel sequencing enable the rapid sequencing of this RNA (RNA-Seq) in a genome-wide manner in order to quantify the amount of synthesized gene product.

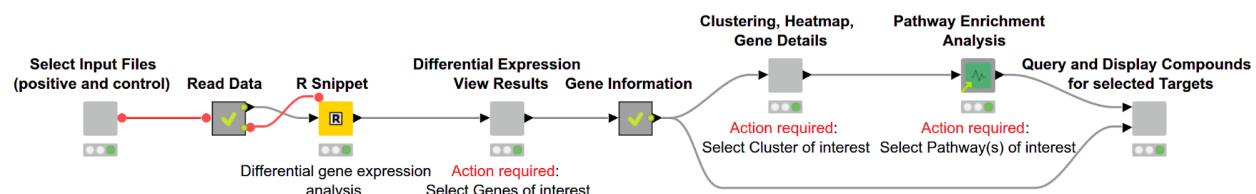


Figure 28 Overview of workflow. Differentially expressed genes are discovered using R and then displayed in an interactive view. Subsequently, genes are hierarchically clustered based on their expression pattern, and the results are shown via a dendrogram alongside a heatmap. We then perform a pathway enrichment analysis and look for compounds targeting the gene product of interest.

In our use case today we:

- Analyze RNA-Seq data from tumors and matched normal tissue from three patients with oral squamous cell carcinomas<sup>1</sup>
- Investigate all statistically significant over/under expressed genes and select interesting ones by looking into their functional annotations.
- Using hierarchical clustering, we select a cluster of similarly expressed genes and investigate their pathway enrichment.
- Search for compounds that target the gene products we picked.

As illustrated in Figure 28, the overall workflow, Gene Expression Analysis, which can be downloaded from the KNIME Hub, consists of the following steps:

1. Input data
2. Find differentially expressed genes
3. View results of differential gene expression analysis
4. Clustering
5. Pathway enrichment
6. Display compounds targeting gene product of interest

The user can select the files containing RNA-Seq data for samples with and without a disease of interest (positive and control, respectively). This data then gets used in the R Snippet to find differentially expressed genes. The user can investigate those genes and select genes of interest based on statistics from the gene expression analysis. We then cluster the genes based on similar expression profiles and investigate their biological pathways. In the last step we search for compounds targeting the selected gene products.

## **Input data**

As mentioned in the previous section, today's example uses RNA-Seq data from normal and tumor cells from patients with oral squamous cell carcinomas<sup>1</sup>. The standard procedure to generate that data consists of the following steps: the RNA of the cells is reversely transcribed to cDNA and then sequenced using massively parallel sequencing resulting in short sequenced reads. Subsequently, these reads are mapped back to the reference genome to identify the genes from which they originated. This results in a count for each position in a gene representing the amount of gene product. In our data set, read counts for 10,542 genes were collected.

## Find differentially expressed genes

One of the strengths of KNIME Analytics Platform lies in its openness for other tools. This allows you to easily harness the power of those tools such as R with all of its libraries. In our case today we want to utilize a commonly used R library for differential expression analysis of RNA-seq expression profiles: [edgeR<sup>3</sup>](#). edgeR implements a range of statistical methods including likelihood tests based on generalized linear models (GLM). GLMs are most commonly used to model binary or count data which makes them perfectly suited to model the aforementioned read counts. [GLMs](#) model a response by a linear function of explanatory variables and allow for constraints such as a restriction on the range of the response Y or the variance of Y depending on the mean. Hence, a generalized linear model is made up of three components: (1) a linear predictor, (2) a link function that describes how the mean depends on the linear predictor and (3) a variance function that describes how the variance depends on the mean [ $\text{var}(Y_i) = \phi V(\mu)$ , with  $\phi$  being the dispersion parameter].

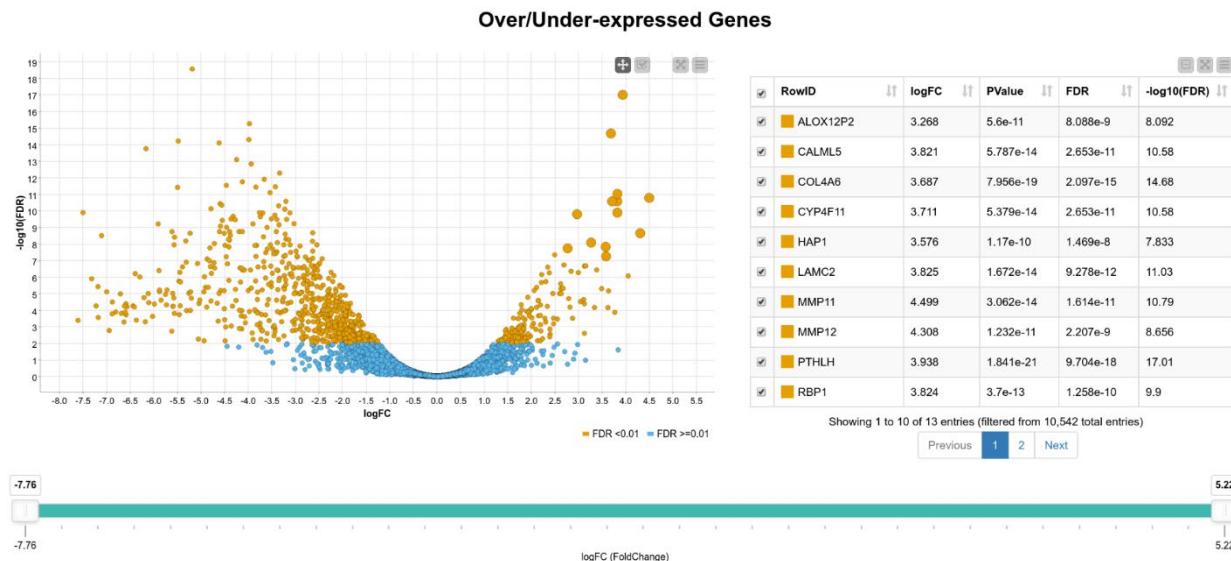


Figure 29 View of differentially expressed genes. A scatter plot of the fold change vs. the FDR and a table with details of the result is shown. The user can select genes in the plot, the table, or filter by fold change using the range slider.

Therefore, in our R snippet, we use the read count data from tumor vs. normal tissue and estimate the dispersion parameter. In the next step, we fit the generalized linear model and apply a likelihood-ratio test. This results in a log fold-change (logFC) and a p-value for each gene. The logFC describes how much a quantity changes between an original and a subsequent measurement. In our case that means how much the read counts per gene differ between tumor and normal cells. As we do this simultaneously for all 10,542 genes, we have to make sure to apply a multiple testing correction<sup>4</sup>. We use the default method provided by edgeR, Benjamini-Hochberg, which has the false discovery rate (FDR) as output.

As we are interested in the relative changes in expression levels between conditions, we do not have to account for factors such as varying gene length. However, we have to account for differing sequencing depth and RNA composition. Sequencing depth is adjusted in edgeR as part

of the basic modeling procedure. To adjust for RNA composition effects, where highly expressed genes can cause the remaining genes to be under-sampled in that sample, we use the function `calcNormFactors`. In addition to the fold-change and the FDR for each gene, we extract the depth-normalized read counts (counts-per-million) for each gene in our analysis.

## **View of differentially expressed genes**

We now have statistics for 10,542 genes, so in the following steps we want to narrow down the results to genes of interest. For that, we create an interactive composite view in which the user can select genes for further analysis. As can be seen in Figure 29, we display the fold-change vs. the FDR on a logarithmic scale in a scatter plot. The colors indicate if the FDR exceeds 0.01 or not. In addition, we show an interactive table with details of the results (FDR,  $-\log_{10}(FDR)$ , p-value, gene name) and a range slider that allows the user to interactively filter for the logFC. We can now, for example, extract only genes that are significantly upregulated in tumor cells by box selecting the genes in the upper right corner of the scatter plot.

## Heatmap and Dendrogram

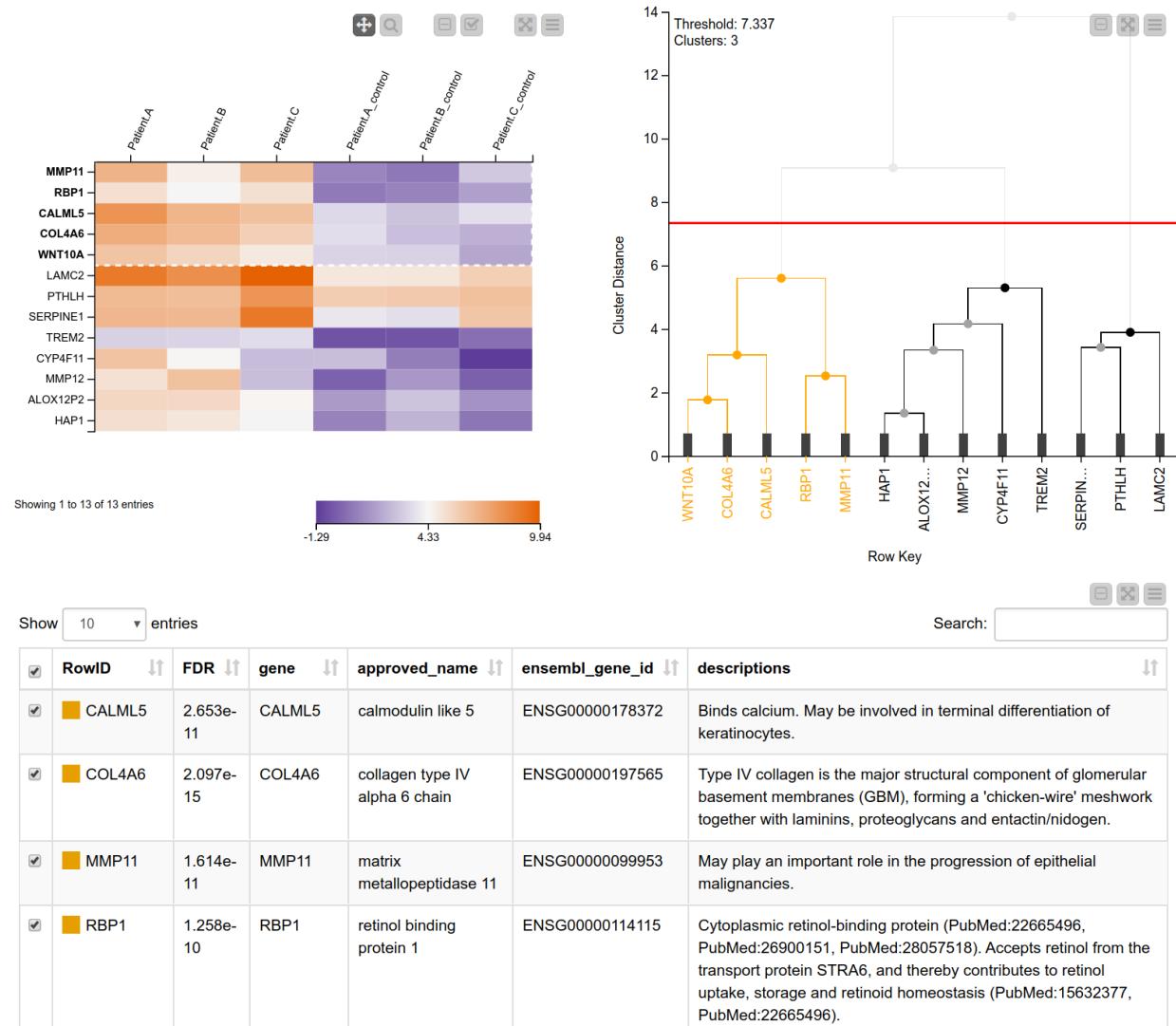


Figure 30 View of heatmap with normalized read counts and dendrogram showing the hierarchical clustering of the counts. The heatmap is sorted according to the clustering. This combination of the heatmap with the dendrogram can be easily achieved using the shared component "Hierarchical Clustering and Heatmap". Additionally, a table with more detailed information is shown.

## Clustering

Having narrowed down our search space, let's have a closer look into the biological processes associated with these genes. Similar expression patterns of genes often point to a common function<sup>5</sup>. To find those genes with similar expression patterns, we perform a hierarchical clustering on the normalized read counts and display the results in a hierarchical cluster tree. For this, we can use a shared component that can be found on the KNIME Hub, the component Hierarchical Clustering and Heatmap. This component allows you to perform a hierarchical clustering on numerical columns of your choice and to display a heatmap sorted according to the clustering results. We combine this component with a table containing more detailed information

about the genes (see Figure 30), allowing us to interactively identify and pick a cluster of interest. In our case we select the one showing high (orange) values in tumor cells and low (blue) values in the matched normal tissue. As we can learn from the details in the table of our composite view, this cluster includes MMP11 (matrix metallopeptidase 11) which may play an important role in the progression of epithelial malignancies, and COL4A6 (collagen type IV alpha 6 chain) which is the major structural component of glomerular basement membranes.

To further investigate shared function of the selected genes we perform a pathway enrichment analysis in the next step.

## Pathway enrichment

A pathway consists of a set of genes related to a specific biological function. As genes are often annotated to a lot of pathways, a pathway enrichment analysis allows us to find those pathways that are enriched in the input set of genes more than would be expected by chance<sup>6</sup>. Pathway enrichment analysis is, therefore, a widely used tool for gaining insight into the underlying biology of differentially expressed genes, as it reduces complexity and has increased explanatory power<sup>7</sup>.

Again, we can use the KNIME Hub to easily drag and drop a shared component called Pathway Enrichment Analysis. This component makes use of the [Reactome](#) Pathway Database, a resource which is open-source, curated and peer-reviewed. It provides a pathway enrichment web service which can be easily accessed by KNIME Analytics Platform. The component takes as input a set of [Ensembl gene IDs](#) and automatically performs the pathway enrichment analysis, the results can be seen in Figure 31. The pathways with the most significant enrichment are “Collagen chain trimerization” and “Degradation of the extracellular matrix”. Both MMP11 and COL4A6 are part of these two pathways. Indeed, collagen is an essential part of the extracellular matrix and extracellular matrix interactions are known to be involved in the process of tumor invasion and metastasis in oral squamous carcinoma<sup>8</sup>.

This further corroborates our hypothesis that these genes play an important role in oral squamous cell carcinomas from our patients’ tumor cells.

## Pathway Enrichment Overview

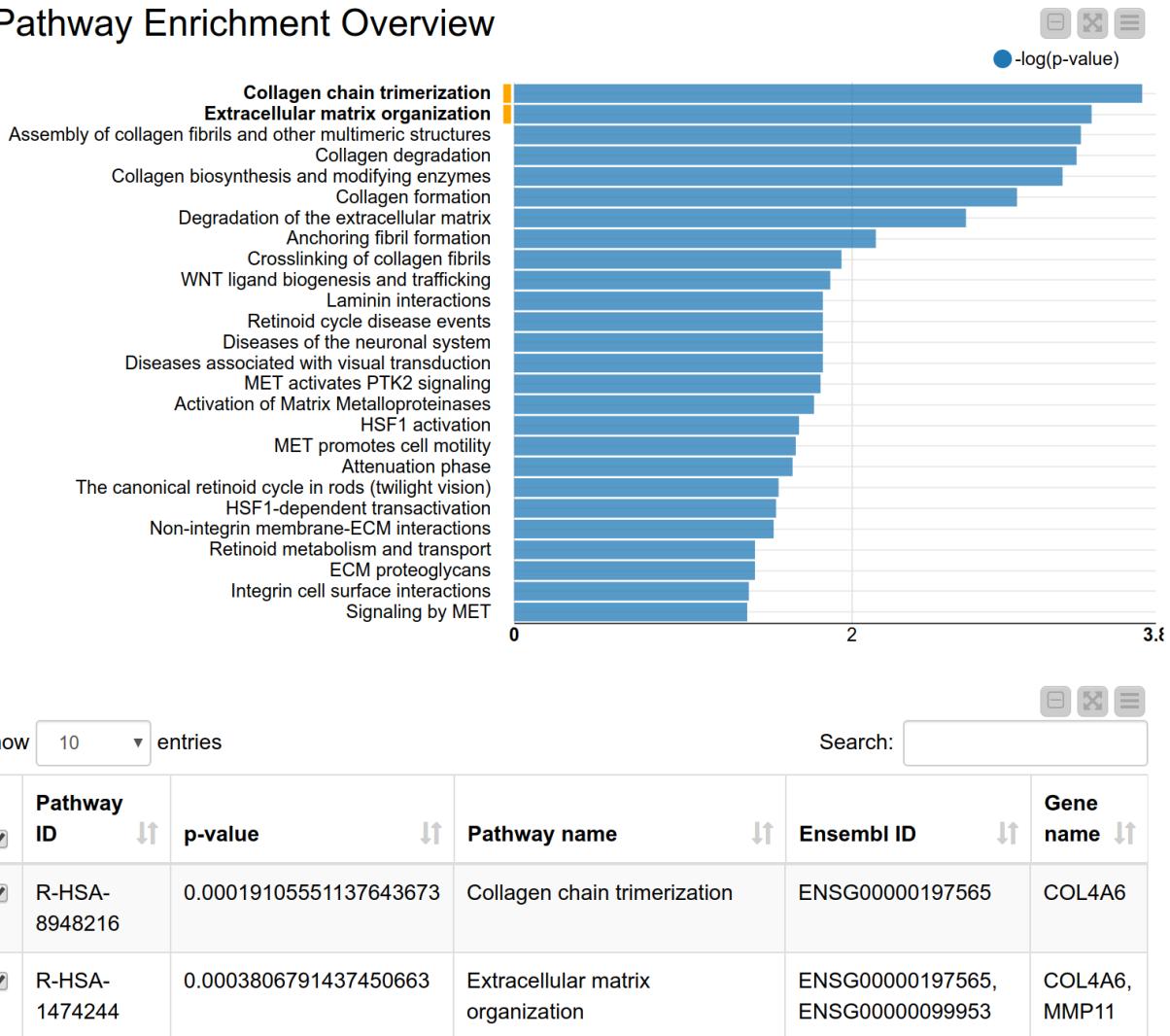


Figure 31 Pathway enrichment view. The pathways with the highest enrichment are “Collagen chain trimerization” and “Degradation of the extracellular matrix”.

## View compounds targeting gene product of interest

In this final step, we want to check if we can possibly interfere with the disease for which we did our expression data analysis. For that, we look for compounds that target the selected gene products. As we have seen in a previous blog article [Interactive exploration and analysis of scientific datasets using Google BigQuery and KNIME Analytics Platform](#), Google BigQuery offers effortless access to public life sciences data. For this, you need to set up a BigQuery Account first, you can find more details on how to do that in this blog article: [Tutorial - Importing Bike Data from GoogleBigquery](#).

In particular, we can easily query bioactivity data from the database ChEMBL using the KNIME Google BigQuery Connector in combination with the KNIME Database nodes. For our query we gather all human synonyms for the gene products of choice and extract compounds known to

target those. This allows us to retrieve information for all those compounds including the name, the assay ID, the type of measurement (e.g. IC<sub>50</sub> or Ki), and the structure as SMILES string.

From the SMILES string we create images of the molecules and display them in a tile view. As can be seen in Figure 32 we found only results for MMP11, Matrix metalloproteinase-11 also known as Stromelysin-3. MMP11 is known to be involved in extracellular matrix breakdown in normal physiological processes and has been implicated in promoting cancer development by inhibiting apoptosis as well as enhancing migration and invasion of cancer cells<sup>9</sup>. Additionally, it has been revealed that MMP-11 expression in oral squamous cell carcinoma samples can predict the progression and the survival of oral squamous cell carcinoma patients<sup>10</sup>. Moreover, MMP11 has recently been identified as potential therapeutic target in lung adenocarcinoma<sup>11</sup>.

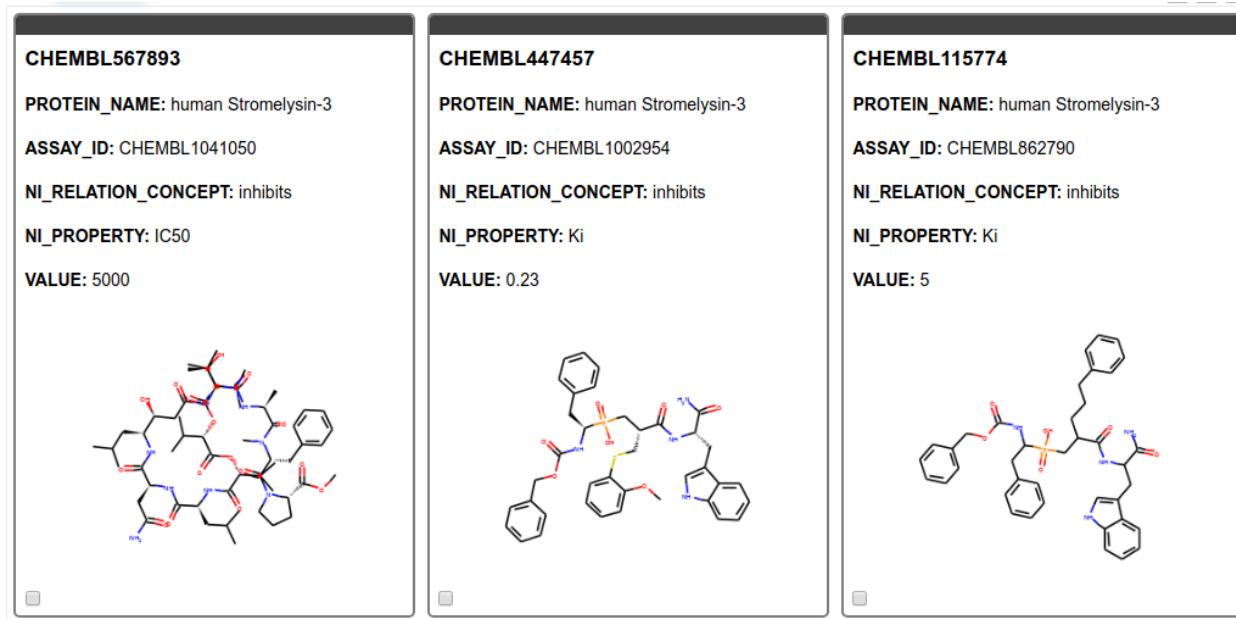


Figure 32 Tile view with compounds targeting the gene product of interest. The ChEMBL ID for the ligand, the name of the target, the assay ID, the action, the type of measurement, and its value are shown.

## Summary

Today we learned how to perform a classic task in bioinformatics: differential gene expression analysis for a disease of interest. We created an interactive view that allowed the user to select significantly under/over expressed genes. From there we further narrowed that set of genes down to genes with similar expression patterns and common function. In the last step, we searched for compounds targeting the discovered gene products thereby offering the possibility to interfere with the disease under investigation. We applied our workflow to data from normal and tumor cells from patients with oral squamous cell carcinoma. Through our analysis we were able to identify a gene that has been independently implicated as a therapeutic target for carcinomas. Moreover, it has been shown that the expression of that gene can predict disease progression as well as survival of oral squamous cell carcinoma patients. All this was facilitated by KNIME's openness for other tools which enabled us to use our favourite R library, extract data from Google's BigQuery and use shared components to customize our analysis.

All steps of the analysis can also be performed on the WebPortal through the interactive views of the components.

## References

1. Tuch, B., Laborde, R., Xu, X., Gu, J., Chung, C., & Monighetti, C. et al. (2010). Tumor Transcriptome Sequencing Reveals Allelic Expression Imbalances Associated with Copy Number Alterations. *Plos ONE*, 5(2), e9317. doi: 10.1371/journal.pone.0009317
3. Robinson, M., McCarthy, D., & Smyth, G. (2009). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1), 139-140. doi: 10.1093/bioinformatics/btp616
4. Noble, W. (2009). How does multiple testing correction work?. *Nature Biotechnology*, 27(12), 1135-1137. doi: 10.1038/nbt1209-1135
5. Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., & Levine, A. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings Of The National Academy Of Sciences*, 96(12), 6745-6750. doi: 10.1073/pnas.96.12.6745
6. Reimand, J., Isserlin, R., Voisin, V. et al. Pathway enrichment analysis and visualization of omics data using g:Profiler, GSEA, Cytoscape and EnrichmentMap. *Nat Protoc* 14, 482–517 (2019). <https://doi.org/10.1038/s41596-018-0103-9>
7. Khatri P, Sirota M, Butte AJ (2012) Ten Years of Pathway Analysis: Current Approaches and Outstanding Challenges. *PLoS Comput Biol* 8(2): e1002375. <https://doi.org/10.1371/journal.pcbi.1002375>
8. Lyons, A., & Jones, J. (2007). Cell adhesion molecules, the extracellular matrix and oral squamous carcinoma. *International Journal Of Oral And Maxillofacial Surgery*, 36(8), 671-679. doi: 10.1016/j.ijom.2007.04.002
9. Zhang, X., Huang, S., Guo, J., Zhou, L., You, L., Zhang, T., & Zhao, Y. (2016). Insights into the distinct roles of MMP-11 in tumor biology and future therapeutics (Review). *International Journal Of Oncology*, 48(5), 1783-1793. doi: 10.3892/ijo.2016.3400
10. Hsin, C., Chou, Y., Yang, S., Su, S., Chuang, Y., Lin, S., & Lin, C. (2017). MMP-11 promoted the oral cancer migration and FAK/Src activation. *Oncotarget*, 8(20). doi: 10.18632/oncotarget.15824
11. Yang, H., Jiang, P., Liu, D., Wang, H., Deng, Q., & Niu, X. et al. (2019). Matrix Metalloproteinase 11 Is a Potential Therapeutic Target in Lung Adenocarcinoma. *Molecular Therapy - Oncolytics*, 14, 82-93. doi: 10.1016/j.omto.2019.03.012

## 2.2 Gut Microbiome Analysis with KNIME Analytics Platform

By Temesgen Dadi

Find the workflow(s) here: <https://kni.me/w/O7Mx7ZakjXbft540>

Microbiomes living inside and on us produce essential enzymes, breakdown nutrients that our body by itself couldn't, train our immune system and are our first line of defense against pathogens. Our health depends on them. This makes qualitative and quantitative analysis of microbiomes an important undertaking. The first step of such analysis is to know which group of microbes are living in a particular body site, such as our gut, and to estimate their respective relative abundances. This step is known as taxonomic profiling.

In this blog, I present a step by step guide how to perform taxonomic profiling on microbial communities using the 16S ribosomal RNA gene as a fingerprint. The data I picked is coming from a study on the dynamics of the gut microbiome during the process of fecal transplant in 10 inflammatory bowel disease (IBD) patients. 16S-rRNA sequences were collected from fecal samples of IBD patients and their donors at different time points of fecal transplant. I will be using the KNIME Analytics Platform and its R-Integration for the whole process. The R-Integration of KNIME allows me to use a domain specific program called DADA2 which is available only as an R package. The blog showcases how to get data directly from the [European Nucleotide Archive](#) via REST and FTP services, pre-process data, use an external R-package within KNIME Analytics Platform, and visualize multiple microbiome composition with the purpose of comparing them.

### Human Gut Microbiome

The human gut, irrespective of its health state, is home to approximately  $10^9$  bacteria and other microorganisms which we collectively refer to as microbes. These microbes are of various sorts and play a wide range of roles in keeping us healthy. But for this (them keeping us healthy and we, well, providing for them) to work, there needs to be a particular composition of different types (species) of microbes. Only then, we can get the essential nutrients just in the right amount, for example. Not having a "good configuration" of the gut microbiome, in other words gut dysbiosis, can cause diseases like Irritable Bowel Syndrome (IBS) and Inflammatory Bowel Disease (IBD)<sup>2,4,5</sup>.

Researchers have been looking at different treatment strategies to alter the microbiome composition towards a state that promotes gut health. The strategies include probiotics, prebiotics, symbiotics and antibiotics<sup>5,8</sup>. While these strategies are still in need of formation to become standard treatment, fecal transplant from a healthy donor to a patient with IBD is gaining momentum as an alternative and blanket solution. The ultimate goal, here, is replacing the gut microbial community of the patient with that of the healthy donor.

To evaluate if the transplant worked or not, one needs to monitor the composition of the gut microbial community before and after transplant. One way of doing that is to take environmental samples, extract all the genetic material from the samples, perform DNA sequencing and use the resulting data to infer

1. which types of organisms are in the sample and
2. what is the prevalence of each type

This is done by using the subtle differences in nucleotide sequences among genomes/genes of different bacterial species. In this blog post, I will focus on using a particular gene, namely the 16S ribosomal RNA gene, for this purpose.

### The 16S ribosomal RNA (16S rRNA) gene

The 16S rRNA gene has the advantage of being highly conserved across almost all prokaryotic species, which facilitates designing primers that can bind to a specific region within the gene 3. The primers are used to selectively perform PCR (Polymerase chain reaction) which produces multiple copies of (parts of) 16S-gene. The resulting sequences are called amplicon sequences. The 16S gene also contains hypervariable regions that can be used as fingerprints to identify the types of bacteria. These variable regions are numbered V1-V9 and have a well-defined locus within the stretch of the gene. In order to identify microbial species/groups, one can use either the entire length of the 16S gene spanning all the variable regions, or part of the 16S gene covering two or three hypervariable regions.

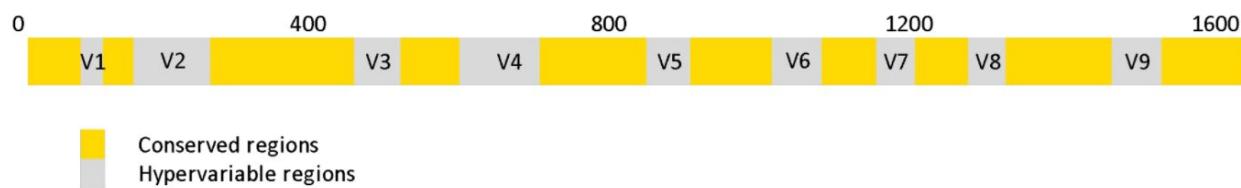


Figure 33 Regions of 16s rRNA genes. The grey regions indicate hypervariable regions that can be used as fingerprints to identify the types of bacteria.

## A KNIME Workflow for Gut Microbiome Analysis of IBD Patients from 16S sequencing data

In this blog post, I present a KNIME workflow created to analyze 16S-rRNA data obtained from the gut of 10 IBD patients at different time points while they undergo fecal transplants. The overall goal is to understand the shift in gut microbiome composition with the help of multiple visualizations.

The workflow uses the DADA2 R package by Callahan et al. 1 to determine the microbial composition from the 16S sequences. This is done via the R-Integration of KNIME which allows usage of such domain specific applications available as R packages with minimal effort. Such

packages are focused on solving a particular scientific problem and are results of months of research and being able to use them directly in a KNIME workflow is just great.

A system wide installation of the DADA2 R package is needed for the workflow to work. Installation instructions can be found [here](#). The main result of DADA2 is a table containing a list of unique amplicon sequences called **Amplicon Sequence Variants (ASV)** and their count. In the workflow, this result will go through further analysis steps using KNIME Analytics Platform to have a dashboard of visualizations of taxonomic profiles across patients and timepoints.

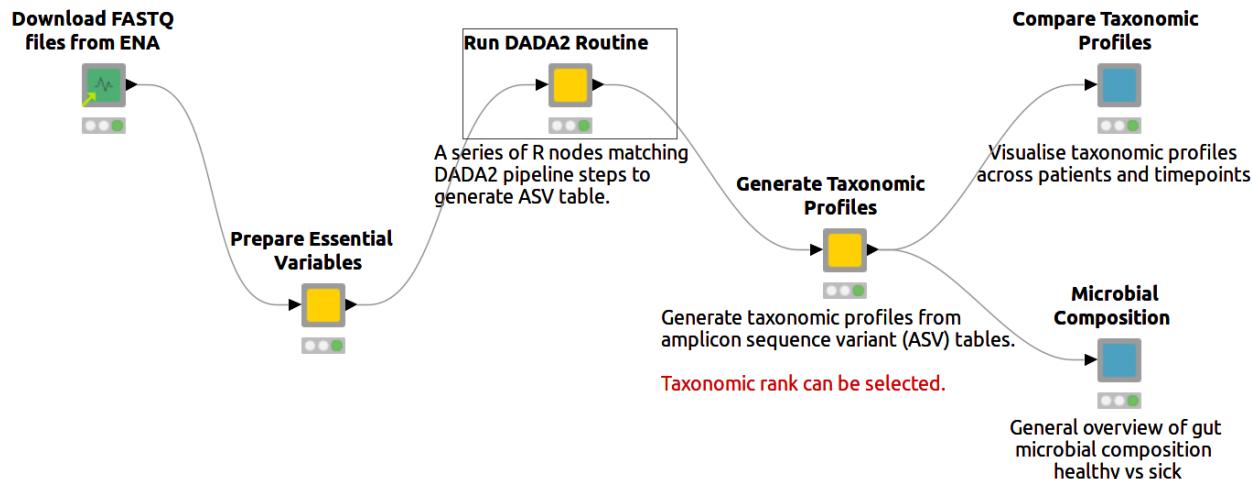


Figure 34 A KNIME workflow for gut microbiome analysis of IBD patients from 16S sequencing data. It can be accessed and downloaded from the KNIME Hub

In short the workflow does the following:

1. Downloads 16S amplicon sequencing files ([FASTQ format](#)) from European Nucleotide Archive (ENA)
2. Uses the R package DADA2 to quality check the sequences and create an amplicon sequence variant table and assign them to a group of bacteria
3. Create taxonomic profile at a desired taxonomic rank
4. Visualize the results to demonstrate the change in the composition of gut microbiota of each patients

Let us dive into each step of the workflow and explain the main ideas behind each part/component of the workflow.

## 1. Download FASTQ sequences from ENA

The first thing to do is getting the DNA sequencing data from European Nucleotide Archive (ENA) where it is publicly available under project identifier **PRJDB4959**. More metadata about the project can be accessed [here](#). I used the “[Download FASTQ files from ENA](#)” component available from the KNIME Hub to easily retrieve our example dataset from the source. The dataset contains a total of 40 FASTQ files representing 10 donors and 10 patients at 3 different time points after going through fecal microbiota transplantation. In each FASTQ file there are thousands of short

DNA sequences (sequencing reads) obtained via amplicon sequencing. The component outputs a table that contains the path to the sequence files of each sample that are downloaded and stored locally.

| Row ID | URI                          | Source URI   | URI |
|--------|------------------------------|--|-----|
| Row0   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065904.fastq.gz; EXT: gz |     |
| Row1   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065905.fastq.gz; EXT: gz |     |
| Row2   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065906.fastq.gz; EXT: gz |     |
| Row3   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065907.fastq.gz; EXT: gz |     |
| Row4   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065908.fastq.gz; EXT: gz |     |
| Row5   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065909.fastq.gz; EXT: gz |     |
| Row6   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065910.fastq.gz; EXT: gz |     |
| Row7   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065911.fastq.gz; EXT: gz |     |
| Row8   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065912.fastq.gz; EXT: gz |     |
| Row9   | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065913.fastq.gz; EXT: gz |     |
| Row10  | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065914.fastq.gz; EXT: gz |     |
| Row11  | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065915.fastq.gz; EXT: gz |     |
| Row12  | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065916.fastq.gz; EXT: gz |     |
| Row13  | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065917.fastq.gz; EXT: gz |     |
| Row14  | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065918.fastq.gz; EXT: gz |     |
| Row15  | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065919.fastq.gz; EXT: gz |     |
| Row16  | URI: ftp://ftp.sra.ebi.ac... | URI: file:/Users/dadi/dev/knime-workspace/data/PRJDB4959/DRR065920.fastq.gz; EXT: gz |     |

Figure 35 The output of “Download FASTQ sequences” component showing a partial list of sequences downloaded from EBI.

## 2. Create an Amplicon Sequence Variant (ASV) table

Since I have our sequencing reads of each sample as individual FASTQ files, I can now go ahead and start the analysis with DADA2. A typical DADA2 pipeline starts by inspecting the quality profiles of the input sequencing reads. The results are then used as a guide to perform error correction on the original sequences to account for sequencing errors. Then sequences are truncated at a length where the quality drops for the majority of the sequences. The details on how exactly this is done can be found in the DADA2 paper (Callahan et al.) and I leave that to the interested reader. The error correction is followed by a series of R Scripting nodes matching each stage of the DADA2 pipeline. Each node mostly performs a singular task by calling a DADA2 routine/function. Figure 36 shows an example code snippet inside the R to R node that filter sequences.

```

R Script
1 # check quality dist to get filtering params
2 # plotQualityProfile(fwd_fastq_paths[1:4])
3 .
4 out <- filterAndTrim(fwd=fwd_fastq_paths, filt=fwd_filt_fastq_paths,
5   truncLen = 340, trimLeft = 0, maxEE = 2, truncQ = 2,
6   compress=TRUE, verbose=TRUE, multithread=TRUE) # On Windows set multithread=FALSE
7

```

Figure 36 An example R code snippet that filters and trims sequences.

It is of course possible to use just a single R to R node with the combined R source code instead of a series of nodes. But, I think the later representation makes both understanding and maintaining the pipeline easier.

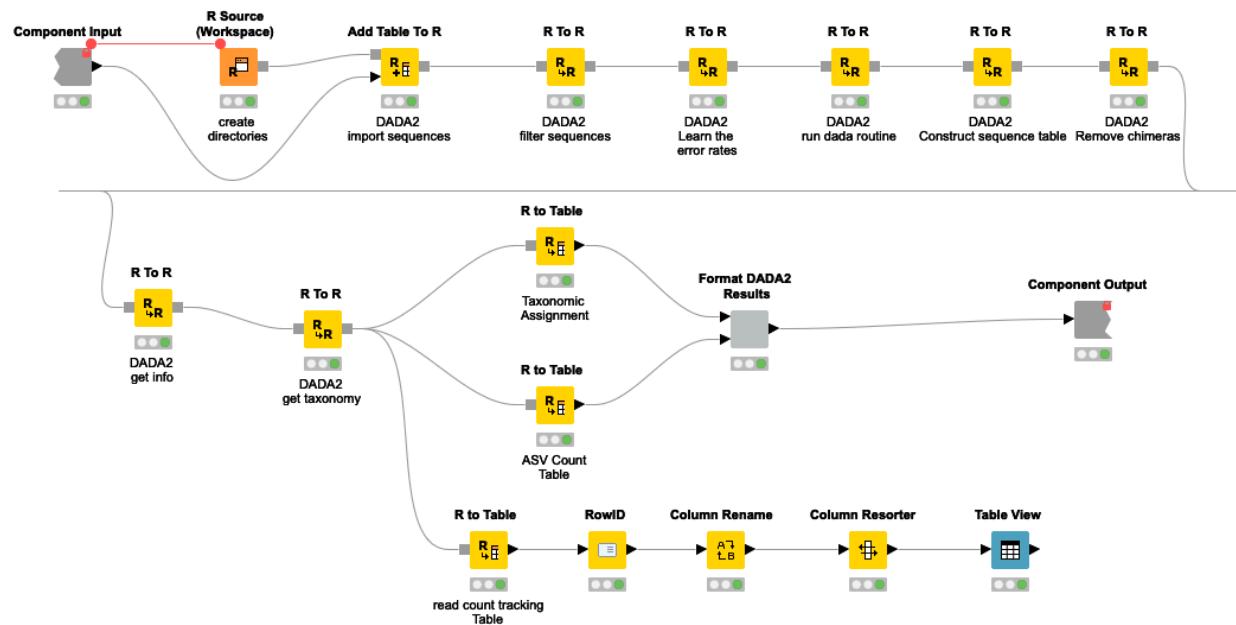


Figure 37 DADA2 pipeline represented by a series of KNIME R Scripting nodes.

The pipeline can be summarized into 5 steps.

1. Sequences below a certain threshold length and quality are filtered out
2. By looking at the error profile, noisy sequences are filtered out. Here probabilistic error correction is done to account for nucleotide differences that are artifacts of the sequencing process.
3. A table of ASVs and their frequency in each sample is generated.

4. Chimeric sequences are removed. Chimeric sequences are sequences that do not exist naturally but created by a faulty PCR process in which sequences from two different origins are artificially concatenated.
5. After getting the ASV count table, the next step is to decode each ASV into a (group of) bacteria/taxa known to be associated with it. I will use a database curated and provided by the authors of DADA2. The database contains a list of known amplicon sequences and the taxa they belong to. Ambiguous sequences are assigned to a more generic taxa. For example, 16S sequences that are equally similar between that of species\_1 and species\_2 will be assigned to a group that covers both species. In this process a given ASV can be assigned to a single bacterial species or at a higher level of taxonomy such as genus or family. This is dependent on the specificity of the ASV.

At the end I will have two tables:

- a) An ASV table where different variants of 16S sequence fragments are represented as rows and samples are represented by columns. The values in the table represent how often a sequence (row) is observed in a sample (column) (Figure 38).

| Data Output - 0:1054:0:857 - R to Table (ASV Count) |        |                              |           |                    |            |                |
|---|--------|------------------------------|-----------|--------------------|------------|----------------|
| File  | Hilite | Navigation                   | View      |                    |            |                |
|   |        | Table "default" – Rows: 1117 |           | Spec – Columns: 40 | Properties | Flow Variables |
| Row ID  |        | DRR065904                    | DRR065905 | DRR065906          | DRR065907  | D              |
| AGGGTTTGATTATGGCTCAGGATGAACGCTGGC...                | 0      | 0                            | 88        | 104                | 79         |                |
| AGGGTTTGATTCTGGCTCAGGATGAACGCTGGC...                | 92     | 26                           | 65        | 0                  | 0          |                |
| AGGGTTTGATTCTGGCTCAGGATGAACGCTGGC...                | 74     | 0                            | 0         | 0                  | 0          |                |
| AGGGTTTGATTATGGCTCAGGATGAACGCTGGC...                | 0      | 0                            | 0         | 0                  | 0          |                |
| AGGGTTTGATTCTGGCTCAGGATGAACGCTGGC...                | 72     | 0                            | 0         | 0                  | 125        |                |
| AGGGTTTGATTCTGGCTCAGGATGAACGCTGGC...                | 142    | 0                            | 52        | 0                  | 0          |                |
| AGGGTTTGATTATGGCTCAGGATGAACGCTGGC...                | 51     | 30                           | 51        | 61                 | 0          |                |
| AGGGTTTGATTCTGGCTCAGGATGAACGCTGGC...                | 0      | 65                           | 45        | 0                  | 107        |                |
| AGGGTTTGATTATGGCTCAGGACGAACGCTGGC...                | 0      | 0                            | 0         | 0                  | 0          |                |
| AGGGTTTGATTCTGGCTCAGGATGAACGCTGGC...                | 117    | 0                            | 63        | 0                  | 0          |                |
| AGGGTTTGATTATGGCTCAGGATGAACGCTGGC...                | 0      | 38                           | 90        | 78                 | 37         |                |
| AGGGTTTGATTCTGGCTCAGGATGAACGCTGGC...                | 0      | 0                            | 68        | 0                  | 0          |                |
| AGGGTTTGATCATGGCTCAGGATGAACGCTGGC...                | 49     | 0                            | 0         | 0                  | 0          |                |
| AGGGTTTGATTCTGGCTCAGGATGAACGCTGGC...                | 0      | 0                            | 0         | 0                  | 0          |                |
| AGGGTTTGATTATGGCTCAGGATGAACGCTGGC...                | 0      | 0                            | 0         | 0                  | 0          |                |

Figure 38 ASV count table. Values in the table show the frequencies of each amplicon sequence variant (rows) in different samples (columns).

- b) The assignment of individual ASVs to a taxonomic entry (Figure 39).

| Port 1 - 0:1054:0:1022 - Format DADA2 Results |             |                              |                |                    |                    |                  |            |        |  |
|---|-------------|------------------------------|----------------|--------------------|--------------------|------------------|------------|--------|--|
| File  | Hilite      | Navigation                   | View           |                    |                    |                  |            |        |  |
|   |             | Table "default" – Rows: 1117 |                | Spec – Columns: 49 | Properties         | Flow Variables   |            |        |  |
| Row ID  | \$ Kingd... | \$ Phylum                    | \$ Class       | \$ Order           | \$ Family          | \$ Genus         | \$ Species | \$ ASV |  |
| c0a03be6...                                   | Bacteria    | Actinobacteria               | Actinobacteria | Bifidobacteriales  | Bifidobacteriaceae | Bifidobacterium  | ?          | AGGGTT |  |
| 7d4b72ee...                                   | Bacteria    | Actinobacteria               | Actinobacteria | Bifidobacteriales  | Bifidobacteriaceae | Bifidobacterium  | ?          | AGGGTT |  |
| edcd0c41f...                                  | Bacteria    | Actinobacteria               | Actinobacteria | Bifidobacteriales  | Bifidobacteriaceae | Bifidobacterium  | ?          | AGGGTT |  |
| 7d193981...                                   | Bacteria    | Firmicutes                   | Clostridia     | Clostridiales      | Lachnospiraceae    | Anaerostipes     | ?          | AGGGTT |  |
| c1a02a8c3...                                  | Bacteria    | Firmicutes                   | Clostridia     | Clostridiales      | Lachnospiraceae    | Blautia          | ?          | AGGGTT |  |
| ec61a20cd...                                  | Bacteria    | Firmicutes                   | Clostridia     | Clostridiales      | Lachnospiraceae    | ?                | ?          | AGGGTT |  |
| c4a4a670...                                   | Bacteria    | Actinobacteria               | Actinobacteria | Bifidobacteriales  | Bifidobacteriaceae | Bifidobacterium  | ?          | AGGGTT |  |
| b1b52e94...                                   | Bacteria    | Actinobacteria               | Actinobacteria | Bifidobacteriales  | Bifidobacteriaceae | Bifidobacterium  | ?          | AGGGTT |  |
| d8083334...                                   | Bacteria    | Firmicutes                   | Clostridia     | Clostridiales      | Clostridiaceae_1   | Clostridium_s... | ?          | AGGGTT |  |
| d4f52722f...                                  | Bacteria    | Actinobacteria               | Actinobacteria | Bifidobacteriales  | Bifidobacteriaceae | Bifidobacterium  | ?          | AGGGTT |  |
| f4486503...                                   | Bacteria    | Actinobacteria               | Actinobacteria | Bifidobacteriales  | Bifidobacteriaceae | Bifidobacterium  | ?          | AGGGTT |  |
| 05513dce...                                   | Bacteria    | Firmicutes                   | Clostridia     | Clostridiales      | Lachnospiraceae    | Anaerostipes     | ?          | AGGGTT |  |
| adf38d51c...                                  | Bacteria    | Firmicutes                   | Clostridia     | Clostridiales      | Lachnospiraceae    | ?                | ?          | AGGGTT |  |
| b9281989...                                   | Bacteria    | Firmicutes                   | Clostridia     | Clostridiales      | Lachnospiraceae    | Blautia          | ?          | AGGGTT |  |

Figure 39 Assignment of individual ASVs to a taxonomic entry.

### Quality control

Before proceeding into joining tables, aggregating and visualizing the results, one needs to check how many of the original sequencing reads made it through each stage of the analysis per

sample. There is a dedicated functionality of the DADA2 package for this purpose. I exposed the result through the Table View node, where the number of sequencing reads that were available originally and how many of them passed different filtration steps are shown. One should look out for an unreasonable reduction of read count, as that could mean a bad sample or wrong combination of parameters in the pipeline.

| SampleID  | Original No. of Reads | After Filteration | After Denoising | After Chimeras Removed |
|-----------|-----------------------|-------------------|-----------------|------------------------|
| DRR065904 | 3000                  | 2607              | 2432            | 2376                   |
| DRR065905 | 3000                  | 2393              | 2260            | 1989                   |
| DRR065906 | 3000                  | 2529              | 2280            | 2123                   |
| DRR065907 | 3000                  | 2367              | 2230            | 1979                   |
| DRR065908 | 3000                  | 2520              | 2375            | 2257                   |
| DRR065909 | 3000                  | 2655              | 2462            | 2288                   |
| DRR065910 | 3000                  | 2355              | 2059            | 1979                   |
| DRR065911 | 3000                  | 2429              | 2175            | 2012                   |
| DRR065912 | 3000                  | 2382              | 2112            | 1868                   |
| DRR065913 | 3000                  | 2625              | 2441            | 2262                   |

Figure 40 Sequencing read statistics. The quality looks good, as there is no unreasonable reduction of read count.

The table containing the analysis statistics looks just fine. Starting from 3000 sequencing reads I ended up with 2376, 1989, and 2123 reads. For the second row the chimera removal step took away about 300 reads which is quite higher compared to the other two displayed here. Chimeric sequences are simply sequences that do not exist naturally but created by a faulty PCR process in which sequences from two different origins are artificially concatenated.

### 3. Create taxonomic profile at a desired taxonomic rank

Depending on the level granularity required or for the purpose of finding more fitting patterns among samples or sample groups it is important to produce taxonomic profiles at different relative levels of grouping (taxonomic ranks). In the workflow, it is possible to select among 7 different ranks. The ranks are Kingdom, Phylum, Class, Order, Family, Genus and Species from generic to specific. Selecting the rank is usually done by looking at the taxonomic assignment and the most specific rank with not too many missing values in the corresponding column is chosen. In our example, this is either genus or family. The counts of sequences will be grouped

by the chosen rank and relative abundance is calculated to show the percentage of each group of that taxonomic rank.

| taxonomic profile - 0:1053 - Generate Taxonomic Profiles |             |               |               |   |                    |           |           |           |           |           |      |  |  |  |  |
|--|-------------|---------------|---------------|---|--------------------|-----------|-----------|-----------|-----------|-----------|------|--|--|--|--|
| File   | Hilite      | Navigation    | View          | Table "default" – Rows: 34 Spec – Columns: 46 Properties Flow Variables |                    |           |           |           |           |           |      |  |  |  |  |
| Row ID   | \$ Kingd... | \$ Phylum     | \$ Class      | \$ Order  | \$ Family          | D DRRO... | D DF |  |  |  |  |
| Lachnospir...  | Bacteria    | Firmicutes    | Clostridia    | Clostridiales   | Lachnospiraceae    | 47.559    | 31.222    | 24.447    | 21.475    | 49.00     |      |  |  |  |  |
| Bifidobacte...   | Bacteria    | Actinobact... | Actinobact... | Bifidobact...   | Bifidobacteriac... | 17.635    | 13.876    | 30.24     | 16.675    | 25.47     |      |  |  |  |  |
| Ruminococ...   | Bacteria    | Firmicutes    | Clostridia    | Clostridiales   | Ruminococcace...   | 17.466    | 2.011     | 26.189    | 10.965    | 17.63     |      |  |  |  |  |
| Bacteroidac...   | Bacteria    | Bacteroid...  | Bacteroidia   | Bacteroida...   | Bacteroidaceae     | 6.439     | 28.054    | 11.776    | 22.991    | 1.817     |      |  |  |  |  |
| Streptococc...   | Bacteria    | Firmicutes    | Bacilli       | Lactobacill...  | Streptococcaceae   | 5.85      | 0.905     | 0.188     | 1.011     | 1.861     |      |  |  |  |  |
| Erysipelotri...  | Bacteria    | Firmicutes    | Erysipelot... | Erysipelot...   | Erysipelotricha... | 1.936     | 2.413     | 1.319     | 2.527     | 1.152     |      |  |  |  |  |
| Enterobacte...   | Bacteria    | Proteobac...  | Gammapr...    | Enterobact...   | Enterobacteri...   | 0         | 0.553     | 0         | 2.425     | 0         |      |  |  |  |  |
| Coriobacter...   | Bacteria    | Actinobact... | Actinobact... | Coriobacte...   | Coriobacteriac...  | 0         | 0         | 0         | 0         | 1.949     |      |  |  |  |  |
| Prevotellac...   | Bacteria    | Bacteroid...  | Bacteroidia   | Bacteroida...   | Prevotellaceae     | 0         | 0         | 0         | 0         | 0         |      |  |  |  |  |
| Lactobacilla...  | Bacteria    | Firmicutes    | Bacilli       | Lactobacill...  | Lactobacillaceae   | 0         | 3.318     | 0.565     | 0.606     | 0         |      |  |  |  |  |
| Acidaminoc...  | Bacteria    | Firmicutes    | Negativicu... | Selenomon...  | Acidaminococc...   | 0         | 6.787     | 0.471     | 16.069    | 0         |      |  |  |  |  |
| Porphyromon...   | Bacteria    | Bacteroid...  | Bacteroidia   | Bacteroida...   | Porphyromona...    | 1.263     | 0         | 0         | 0.606     | 0.576     |      |  |  |  |  |
| Enterococc...  | Bacteria    | Firmicutes    | Bacilli       | Lactobacill...  | Enterococcaceae    | 0         | 3.67      | 0         | 3.537     | 0         |      |  |  |  |  |
| Veillonellac...  | Bacteria    | Firmicutes    | Negativicu... | Selenomon...  | Veillonellaceae    | 1.178     | 1.659     | 2.073     | 0         | 0         |      |  |  |  |  |
| Clostridiace...  | Bacteria    | Firmicutes    | Clostridia    | Clostridiales   | Clostridiaceae_1   | 0         | 0         | 0         | 0         | 0         |      |  |  |  |  |
| Sutterellacea...   | Bacteria    | Proteobac...  | Betaprote...  | Burkholde...  | Sutterellaceae     | 0.463     | 5.229     | 0.942     | 1.011     | 0.354     |      |  |  |  |  |

Figure 41 Relative abundance table at family level of the taxonomy.

## 4. Visualize the results

It is natural to ask which types of Bacteria are found in a human gut. Using an interactive sunburst chart, I can visualise which group of bacteria are common and their prevalence in the human gut (see Figure 42). The charts represent aggregated/averaged relative abundance of bacterial groups at different taxonomic levels from healthy donors (left) and IBD patients before a fecal transplant (right). Clicking on each portion of the charts displays the actual of sequences that are tied to the selected group and their count in a table view.

A simple [BLAST](#) of the sequences on NCBI can verify the sanity of the method by checking if the sequences are actually related to the group they are assigned to by the pipeline.

The commensal (more common) bacterial families are Lachnospiraceae, Bifidobacteriaceae, Ruminococcaceae and Bacteroidaceae. In general, these groups are higher in abundance in the healthy patients. If we take the currently selected group Lachnospiraceae for example, it is higher in the donors than in patients. These observations are in-line with literature which suggests that IBD patients are characterized by a lower abundance of Bacteroidetes and Lachnospiraceae compared to healthy controls<sup>4</sup>.

The final step of our workflow produces a JavaScript visualization whereby the shift in microbial composition of individual patients is represented as a dashboard of stacked bar plots. The right most stacked bar is always the microbial composition of the donor's gut, whereas the other three bars to the right represent the gut microbiome of the receiver at different time points.

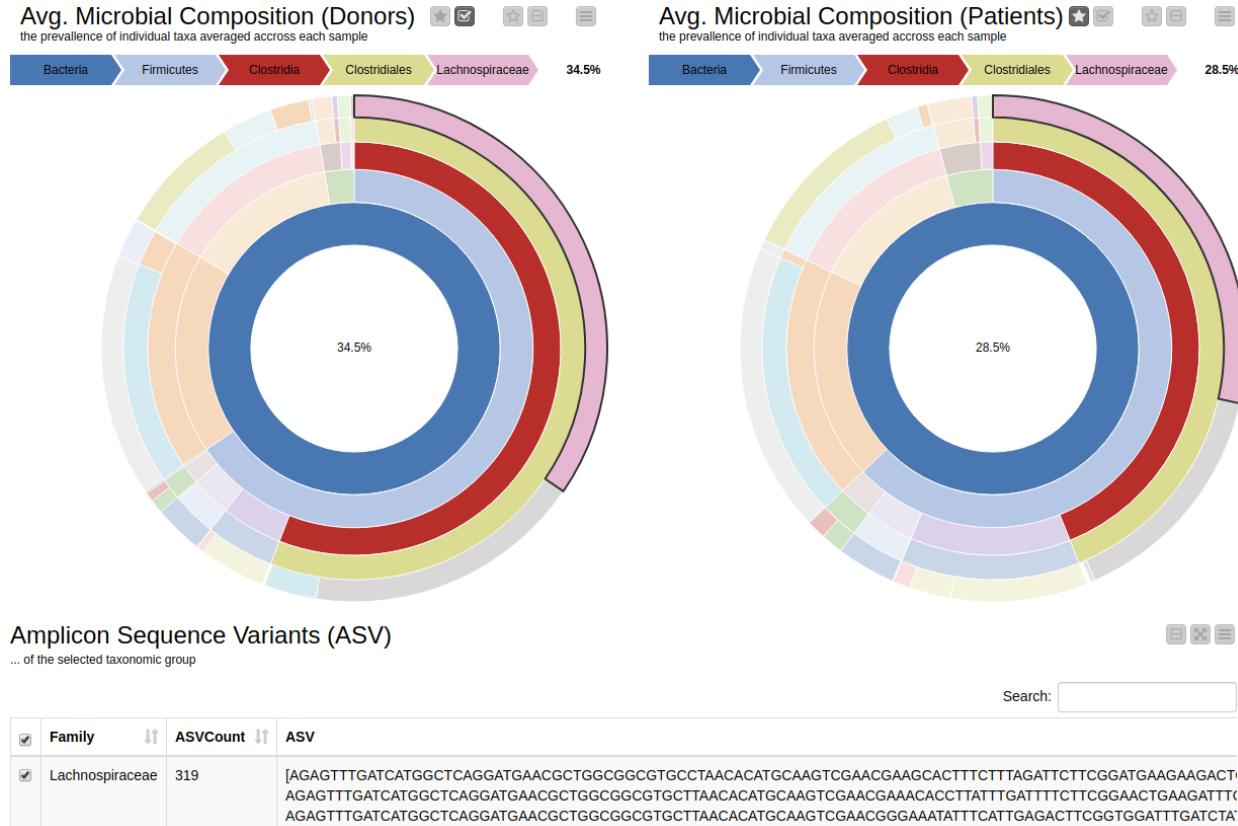


Figure 42 Sunburst charts showing the average composition of the gut microbiome in healthy donors (left) and IBD patients before transplant (right). Selecting a group will show the corresponding sequences representing that group of bacteria in the samples.

First of all, it is interesting to note that the composition of gut microbiomes differ among individuals. This is true even in the healthy donors. Secondly, the patients' gut microbiomes showed some changes towards those of the donors', although in some cases, these changes didn't persist overtime for all patients. For example, the bacterial family Ruminococcaceae (Orange) was present in high abundance in the donor's gut but not so in patient A. But after a week from the transplant it also became as abundant in the patient. Similarly, in patient D the proportion of the bacterial families Streptococcaceae (green) and Enterobacteriaceae (red) were absent in the beginning. But after a week from the transplant these groups of bacteria cover a large proportion of the patient's gut microbiome, as they did in the patient's respective donor.

Let us take a closer look at patient H. From the bar plot it is clear that in the beginning patient H has close to zero Bifidobacteriaceae family in his gut. On the contrary the Donor has an abundance of the same family of bacteria. After the fecal transplant it can be seen that the abundance of this group of bacteria increased significantly. And it is known from literature [6,7] that probiotics rich in Bifidobacteria are successfully used in treating patients with inflammatory bowel diseases.

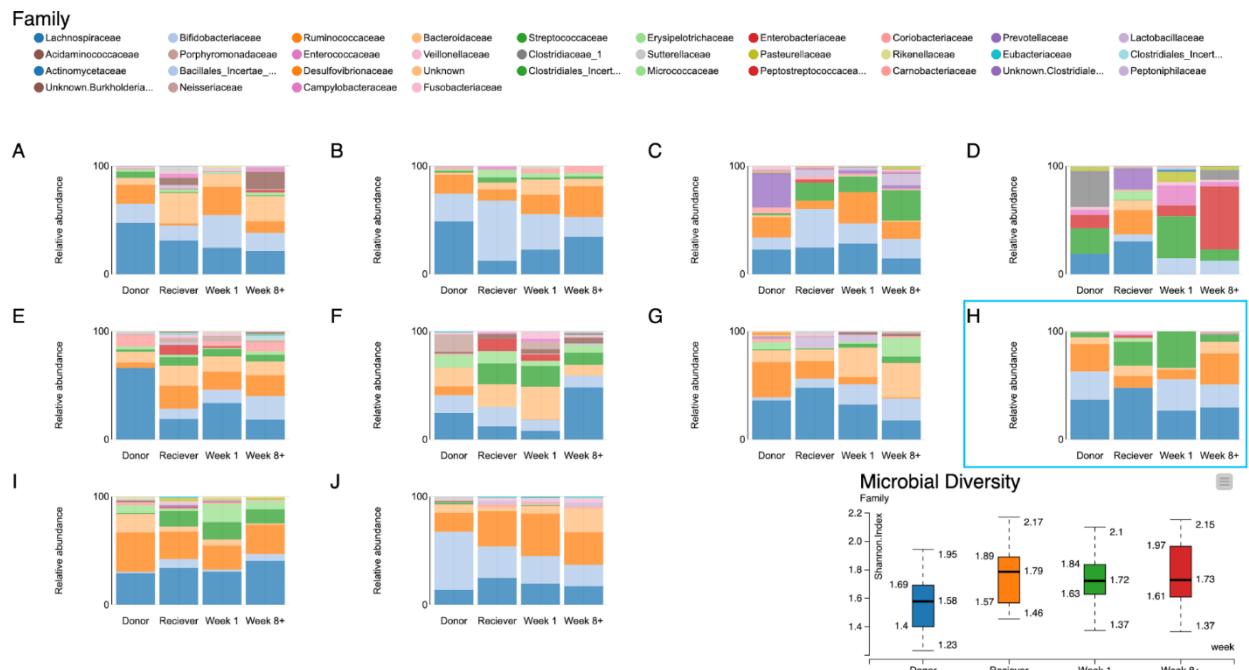


Figure 43 A dashboard of visualizations of taxonomic profiles across 10 patients + their donors (A-J) through different timepoints.

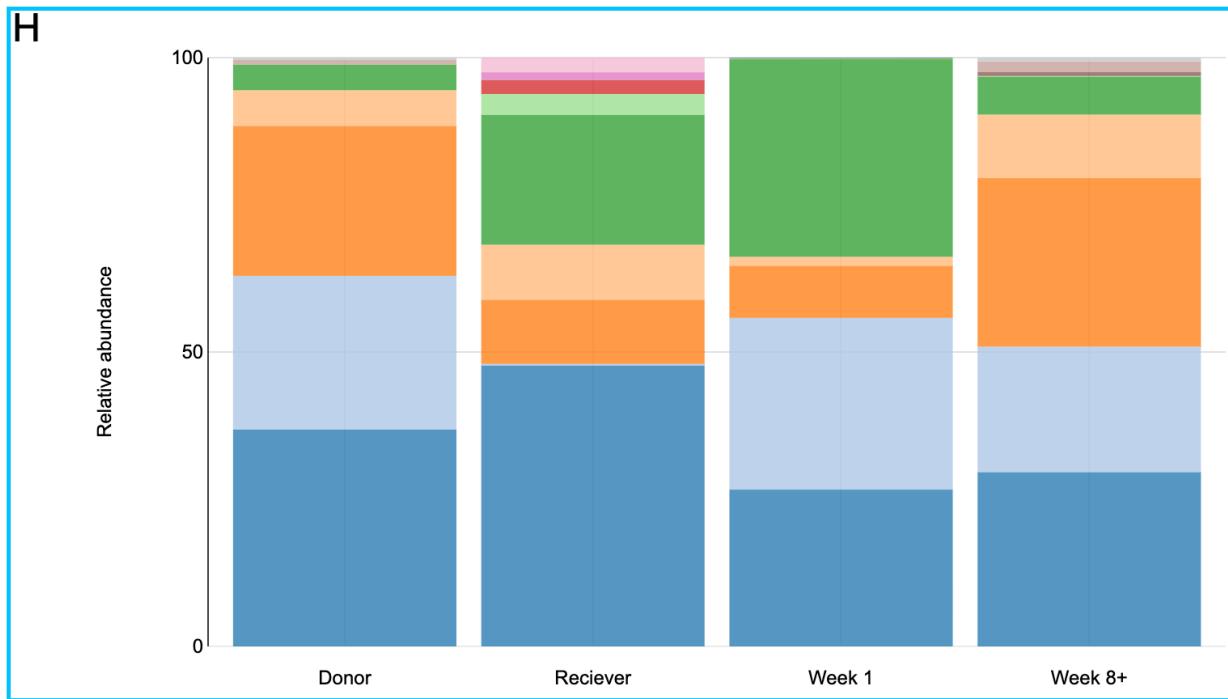


Figure 44 A closer look at the taxonomic profiles of patient J and his/her donor

## Side Note

I can not confirm if the patient's health has improved due to the fecal transplant since I don't have the full treatment data available. I only have the sequencing data with the patient ID and time points at our disposal. It is also unclear if these positive changes are long lasting as the study is limited to 8-12 weeks. I am also fully aware that this is not even close to enough for making scientific conclusions. The focus here is to demonstrate how a KNIME workflow can be developed and used to assist a comparative study of microbial communities at different time points.

## Summary

We have created a KNIME workflow that retrieves sequencing data from public repositories, analyses them, and creates useful visualisations to investigate the changes in microbial composition of patients' gut. We have learned how we can use 16S rRNA amplicon sequences for characterisation of microbial communities in KNIME Analytics Platform. Most importantly, we showed how we can integrate complex and domain specific external R packages in KNIME to create workflows that are transparent and easy to understand but yet powerful enough to get the job done.

The workflow can be used for analysing microbial communities from any other sources. The use cases could range from soil microbiome analysis to monitor the fertility of a soil to environmental bioremediation where microorganisms are used to clean up environmental messes such as oil spills.

## References

1. Callahan BJ, McMurdie PJ, Rosen MJ, Han AW, Johnson AJ, Holmes SP. DADA2: High-resolution sample inference from Illumina amplicon data. *Nat Methods*. 2016;13(7):581–583. doi:10.1038/nmeth.3869
2. Ruairi Robertson, 'Why the Gut Microbiome Is Crucial for Your Health', [www.healthline.com](http://www.healthline.com), June 27, 2017, accessed Feb 2020, <https://www.healthline.com/nutrition/gut-microbiome-and-health>
3. Wang, Y., & Qian, P. Y. (2009). Conservative fragments in bacterial 16S rRNA genes and primer design for 16S ribosomal DNA amplicons in metagenomic studies. *PLoS ONE*, 4(10). <https://doi.org/10.1371/journal.pone.0007401>
4. Kennedy PJ, Cryan JF, Dinan TG, Clarke G. Irritable bowel syndrome: a microbiome-gut-brain axis disorder?. *World J Gastroenterol*. 2014;20(39):14105–14125. doi:10.3748/wjg.v20.i39.14105
5. Distrutti E, Monaldi L, Ricci P, Fiorucci S. Gut microbiota role in irritable bowel syndrome: New therapeutic strategies. *World J Gastroenterol*. 2016;22(7):2219–2241. doi:10.3748/wjg.v22.i7.2219

6. Halfvarson J, Brislawn CJ, Lamendella R, et al. Dynamics of the human gut microbiome in inflammatory bowel disease. *Nat Microbiol.* 2017;2:17004. Published 2017 Feb 13. doi:10.1038/nmicrobiol.2017.4
7. Pozuelo M, Panda S, Santiago A, et al. Reduction of butyrate- and methane-producing microorganisms in patients with Irritable Bowel Syndrome. *Sci Rep.* 2015;5:12693. Published 2015 Aug 4. doi:10.1038/srep12693
8. McFarland LV, Dublin S. Meta-analysis of probiotics for the treatment of irritable bowel syndrome. *World J Gastroenterol.* 2008;14(17):2650–2661. doi:10.3748/wjg.14.2650

## 2.3 Variant Prioritization - Reproducible Workflow with Domain Expert Interaction

By Jeany Prinz

Find the workflow(s) here: <https://kni.me/w/xpPi0YzQ9jwpUC8N>

One of the main features of KNIME Analytics Platform is the capability to build reproducible workflows where it is easy to understand what is going on even if you have not created the workflow yourself or if you have not looked at it for some time. This facilitates sharing your work or running an analysis with different data sets multiple times with only one click. As a bioinformatician, I often use different command line tools with particular parameters in a particular sequence. Using KNIME, I can easily create a pipeline of those tools which not only saves me a lot of time, it also ensures quality, as I can simply look at the workflow and verify that I used the correct parameters. Additionally, I can combine this with other built-in functionality in KNIME such as creating interactive visualizations to investigate the results.

With access to KNIME Server, the possibilities expand even further. I can make the results of my analysis - with interactive interaction points - available to the domain expert via the KNIME WebPortal. That domain expert, in turn, needs not to know anything about KNIME or the processes running under the hood. She or he can simply interact with the data through web pages.

We will create a reproducible workflow for a typical bioinformatics application: variant prioritization. For that, we will combine classic command line tools with built-in functionality in KNIME Analytics Platform including shared components, REST Services, and interactive visualizations. Moreover, we will make the results available for interactive investigation as a Web Application via the KNIME WebPortal.

### Variant Prioritization

A variant refers to a specific region of the genome that differs between two genomes. This can either be a single base, or even big chunks of thousands of bases (insertions/deletions). Analyzing and interpreting those variants can lead to improved patient care, surveillance, and treatment outcomes<sup>1</sup>. As we have demonstrated in previous blog posts about [Sequence Motifs and Mutations](#) and Gene Expression Data Analysis (see chapter 2.1), this variation yields different effects of varying impact. Some lead to no observable effect while others can cause severe genetic diseases. Finding disease-causal variants among large numbers of variants in a human genome remains a major challenge in next-generation sequencing data analysis. Because of the complexity, Cooper (2011) refers to this as a search for "needles in stacks of needles"<sup>2</sup>. Filtering and annotating the variants from such experiments is an important step towards understanding the functional consequences of variants.

One of the most frequently used formats to store variant information is the Variant Call Format ([VCF](#)). The VCF format is specifically designed to store complex genetic variation data in an efficient and concise way. Due to its condensed structure, mining those files is not a straightforward task, and there are several command line tools for filtering and querying information in VCF files such as [VCFtools](#)<sup>3</sup> and [BCFtools](#). Moreover, tools such as the Variant Effect Predictor (VEP)<sup>4</sup> are used for annotation and prioritization of genomic variants.

The VCF file<sup>5</sup> we are using today is based on data from the [1000 genomes project](#), to which we additionally added a known mutation in the [HBB](#) gene leading to a recessive disease. Our ultimate goal is to distinguish disease-causal variants from non-causal ones in our VCF file. The variant prioritization procedure requires biological/biomedical reasoning, and biologists and clinicians are increasingly motivated to handle the task themselves<sup>6</sup>. Therefore, we make the results available as an easily accessible web application through the KNIME WebPortal where the domain expert can interact with the data and further annotate the results.

## Variant Prioritization Workflow

With KNIME Analytics Platform, it is very easy to combine different tools and sources of information. Today, we combine classic command line tools with built-in functionalities in KNIME Analytics Platform including shared components and interactive visualizations in one reproducible workflow.

The Reproducible Variant Prioritization workflow (Figure 45) consists of three main steps:

1. Filter Variants using the command line tools Bcftools and VCFtools
2. Predict Variant Effects through a shared component using Ensembl's Variant Effect Predictor (VEP) via their REST API
3. Create an interactive view through which the domain expert can then filter and manually annotate variants

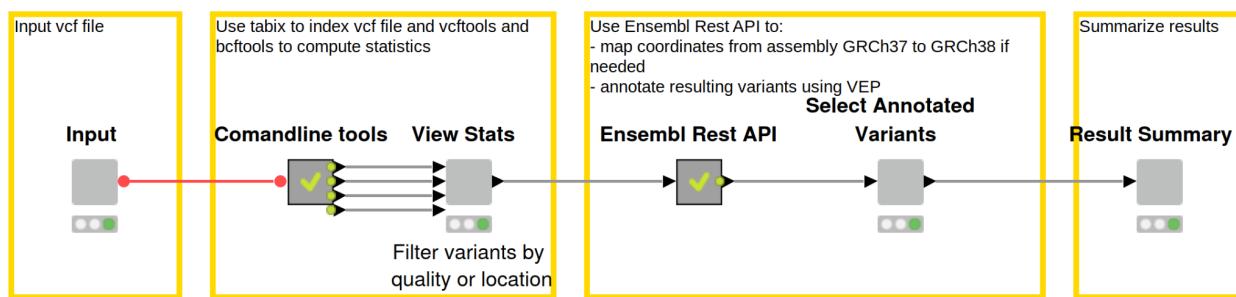


Figure 45 Reproducible workflow for variant prioritization with domain expert interaction. First, the data is filtered using common command line tools, and statistics about the data are displayed in an interactive view. Effects of the variants are subsequently predicted using Ensembl's REST API, and results are made available to the domain expert for review and selection. Lastly, final results are summarized including the intermediate files, the final result file, and all commands used throughout the workflow. Download this workflow from the KNIME Hub here.

Using the KNIME WebPortal, the results of each step can be easily accessed through web pages. The domain expert can interact with the data without having to worry about the tools under the hood. Furthermore, classic bioinformatics command line tools typically run on a Linux environment and through the KNIME WebPortal one can easily access the workflow from Windows.

With KNIME Server 4.11.0, we redesigned the WebPortal frontend with a focus on user interface and user experience. In the following we will use the new WebPortal to make our workflow accessible to the user with domain expertise but not necessarily knowledge of command lines or KNIME.

We start with a VCF file that the user uploads. The user also has to specify the output directory and a prefix for the result files. When running the workflow on your local Linux machine, the output directory is a local path. When running it on the Server, it should point to a directory on the Server. All results (intermediate and final) will be stored in the specified directory and start with the given prefix. In addition, the user should select the [assembly](#) that was used to create the data from a drop-down menu. Finally, the user must decide if Insertions/Deletions (Indels) and heterozygous variants should be removed. For many use cases, it is good practice to analyze Indels separately. Heterozygous variants can be removed if we are investigating, for example, a recessive genetic disorder. In our case, the data is based on coordinates from the [GRCh37](#) assembly version and we remove Indels and heterozygous variants.

These inputs are then used to create command line commands, which are utilized to compute statistics of the data and to filter it by different criteria in the following steps:

1. To enhance performance, we use bgzip and tabix to compress and index data
2. We use BCFtools to compute basic statistics about the data including the number of heterozygous and homozygous variants.
3. Depending on the user input we filter the data in two steps:
  - We keep only variants that are homozygous using BCFtools if that option was selected in the beginning.
  - If selected by the user, we remove Insertions/Deletions (Indels) using VCFtools.
4. We use VCFtools to extract the Phred quality scores. The [Phred Score](#) is logarithmically related to the base-calling error probabilities. For example, if Phred assigns a quality score of 30 to a base, the chances that this base is called incorrectly are 1 in 1000.

The specific commands used are also summarized in the last step of the workflow (see Figure 45). In order to run the workflow, you need to install the relevant command line tools.

We display the results in an interactive view, as can be found in Figure 46. The user can see the number of homozygous and heterozygous variants for the sample under investigation in the table

view. In the parallel coordinates plot, one can select a region of interest and exclude/include variants based on their quality. Making a selection based on domain expertise will also decrease the runtime of the next step, which can take a while as there are REST calls involved. If an error occurs in the command line tools, we display those at the bottom of the interactive view.

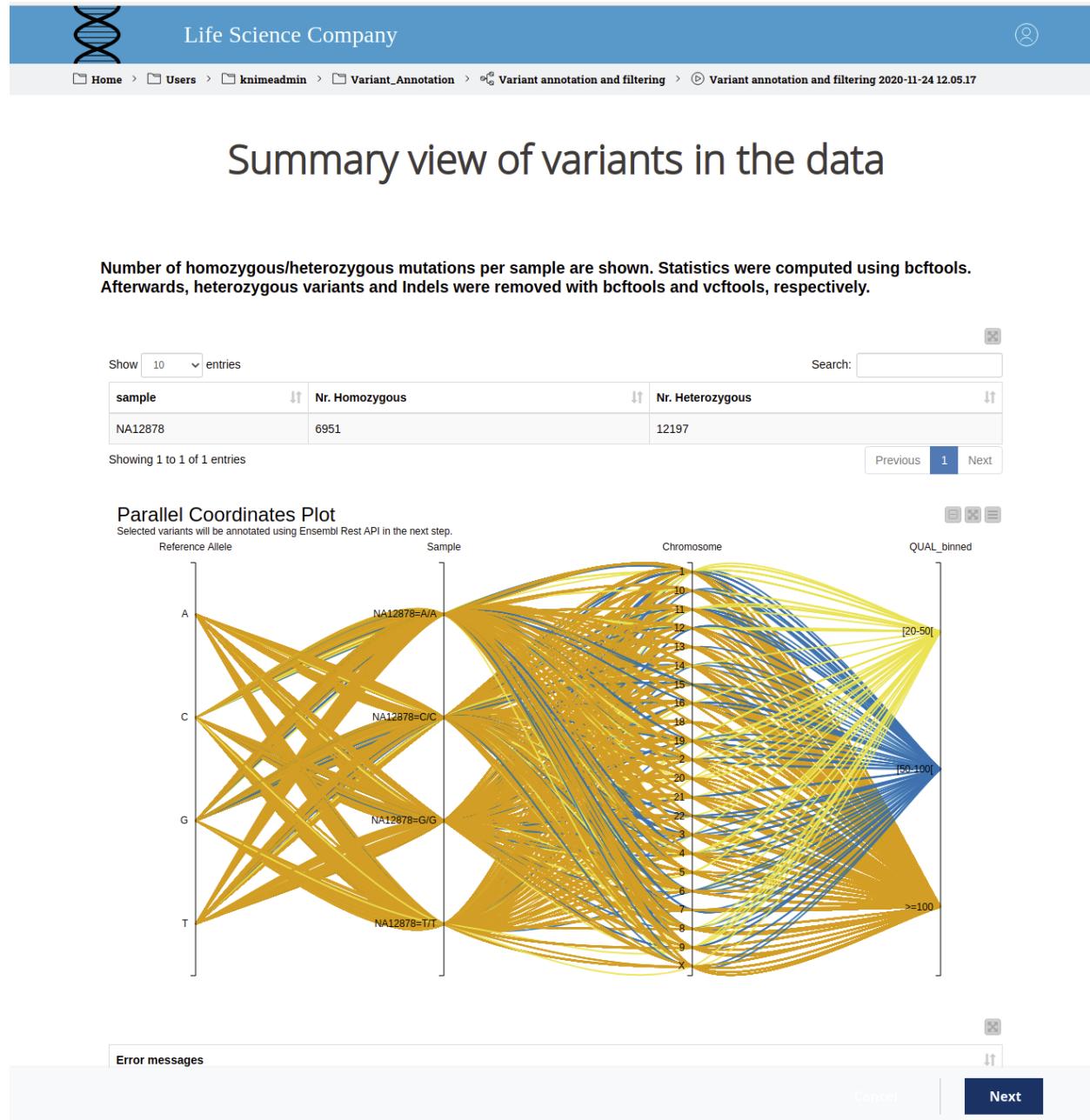


Figure 46 WebPortal view of variants in the data. The sample ID and the number of homozygous and heterozygous variants are displayed in a table. In the parallel coordinates plot, the reference allele, the sample, and the deviating allele, the region and the binned quality score are shown. The colors indicate the binned quality score. The user can select a region of interest and exclude/include variants based on their quality.

As mentioned above, variants yield different effects of varying impact. Variants that alter the amino acid sequence for which they encode or variants located in regulatory regions often have a stronger impact than variants that reside, for example, in intergenic regions. Those differences can be used to predict the influence of a variant. In the next step, we do exactly this with the Variant Effect Predictor (VEP) from Ensembl. Instead of using the command line, we now retrieve information from Ensembl's REST API. In particular, we employ two GET Requests in order to:

## Convert coordinates from one assembly to another

Converting coordinates from one genome assembly version to another is a common task in bioinformatics. In our example, our initial VCF is based on coordinates from the GRCh37 assembly version. However, the current version that we want to use in the next step is [GRCh38](#). Hence, we created a shared verified component based on [Ensembl's REST API](#) that converts the coordinates between different assembly versions of the human genome. The component can be accessed via the KNIME Hub.

## Retrieve variant annotations from Variant Effect Predictor (VEP)

We determine the effect of genetic variants using Ensembl's Variant Effect Predictor ([VEP](#)). To make this functionality easily accessible, we created a verified shared component, you can find it on the KNIME Hub.

The retrieved annotations include the most severe predicted effect of the variant (most severe consequence), which we display in an interactive table view (see Figure 48). Those consequences are calculated per transcript, so we include the transcript information as well. We complete the created view by adding an explanatory image describing the different consequences (Figure 47):

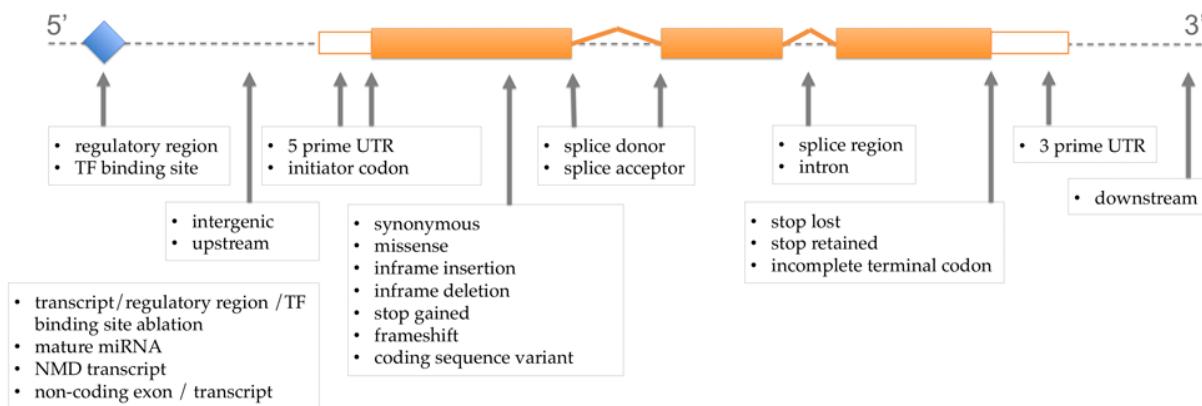


Figure 47 Explanatory Image of Variant Consequences. This image is taken from [Ensembl](#) and illustrates the various impacts of variants depending on where they occur in the genome. The terms used are based on the [Sequence Ontology](#) (SO) for the variation consequences.

Additionally, we extract the allele frequency in the 1000 genomes project from the Ensembl REST API. The minor allele is the less common allele for a single nucleotide polymorphism (SNP), the frequency gives you an idea of how common a SNP is. Filtering for this frequency is motivated by the fact that common variants are less likely to be disease-causal. Therefore, we introduce a

range slider which enables the user to interactively filter by minor allele frequency in the 1000 genomes project.

Furthermore, a recent project analyzed 125,748 [exomes](#) and 15,708 genomes from human sequencing studies and aggregated the data into the Genome Aggregation Database (gnomAD)<sup>7</sup>. We add an additional range slider where the user can filter for the frequencies in this data set as well.

The screenshot shows the Ensembl REST API interface with the following details:

- Header:** Life Science Company, Home > Users > knimeadmin > Variant\_Annotation > Variant annotation and filtering > Variant annotation and filtering 2020-11-24 13.08.30
- Title:** Ensembl REST API
- Description:** Results from the Ensembl REST API Web service extracting Ensembl data and annotations of the Ensembl Variant Effect Predictor VEP.
- Table Headers:** Sample, Reference Allele, Chromosome, start\_GRCh37, start\_GRCh38, gene symbol, minor allele frequency 1000Genomes, most severe consequence, gnomad, sift score, transcript\_id
- Table Data:** A list of variants for sample NA12878, including their position, gene symbol (HRNR, HBB, AC104389.1, OR10A4, ABCC1), and predicted effects (missense\_variant). Each row also includes a checkbox for manual annotation and a link to the transcript ID.
- Footnote:** \*minor allele freq 1000Genomes: global minor allele frequency (calculated using all 1000 Genomes Phase 3 data for this SNP, across populations)
- Footnote:** \*GnomAD: allele frequencies Genome Aggregation Database (GnomAD), <https://gnomad.broadinstitute.org/>
- Footnote:** \*most\_severe\_consequence: most severe predicted effect of the variant [see image below]
- Range Sliders:** Three horizontal sliders at the bottom allow filtering for Sift Score (0.00 to 1.00), minor allele freq 1000Genomes (0.00 to 0.50), and minor allele freq GnomAD (0.00 to 1.00).
- Navigation:** Back, Next, and a search bar.

Figure 48 . Interactive view to filter variants. The table displays the position and the most severe predicted effect of the variant. The gene symbol and the transcript ID are also displayed and the domain expert can add a manual annotation. The range sliders allow filtering for the SIFT score as well as for allele frequencies in the 1000 genomes project and gnomAD.

Life Science Company

Variant annotation and filtering Variant annotation and filtering 2020-11-24 13.08.30

### Summary of Commands and Result Files

**Final Output:**

The final annotation file can be found at: /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles\_new/NewVCF\_finalAnnotations.csv

| Sample      | Reference Allele | Chromosome | start_GRCh37 | start_GRCh38 | e |
|-------------|------------------|------------|--------------|--------------|---|
| NA12878=T/T | C                | 1          | 152186099    | 152213623    | 1 |
| NA12878=A/A | T                | 11         | 5248232      | 5227002      | 5 |
| NA12878=A/A | T                | 11         | 5248232      | 5227002      | 5 |
| NA12878=A/A | T                | 11         | 5248232      | 5227002      | 5 |
| NA12878=A/A | T                | 11         | 5248232      | 5227002      | 5 |

Showing 1 to 5 of 12 entries

**Summary Sunburst**  
Variant consequences per gene/transcript

**Bash Commands**

| Tool         | Description   | Command   | ResultFile   |
|--------------|---|---|--|
| Bgzip, Tabix | Command to compress and index data                            | bgzip -c /home/jeany/workspace/Projects/Vcftools/Data/Spiked_FGG/NA12878.recessive.vcf> /home/jeany/workspace/Projects/Vcftools/Data/Spiked_FGG/NA12878.recessive.vcf.gz; tabix -f /home/jeany/workspace/Projects/Vcftools/Data/Spiked_FGG/NA12878.recessive.vcf.gz | /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_recess |
| BCFTools     | Command to extract variant statistics per sample via bcftools | bcftools stats -s- /home/jeany/workspace/Projects/Vcftools/Data/Spiked_FGG/NA12878.recessive.vcf.gz   grep PSC   cut -f3,5,6,7,8 > /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_stats.tsv  | /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_       |
| BCFTools     | If selected, we keep only homozygous variants                 | bcftools view -g ^het /home/jeany/workspace/Projects/Vcftools/Data/Spiked_FGG/NA12878.recessive.vcf > /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_filterHet.vcf   | /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_       |
| VCFtools     | Command to remove Insertions and Deletions                    | vcftools --vcf /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_filterHet.vcf --remove-indels --max-missing 1 --recode --recode-INFO-all --out /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_rmindels                  | /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_       |
| VCFtools     | Command to extract quality values                             | vcf-query /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_rmindels.recode.vcf -I %CHROM:%POS:%REF [%SAMPLE=%GT] %QUAL > /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_quality.tsv                                     | /home/jeany/workspace/Projects/Vcftools/ResultMultVCFFiles_new/NewVCF_       |

Showing 1 to 5 of 5 entries

[Back](#) [Delete this result](#) [Close](#)

Figure 49 Summary of commands and result files. The table displays the final prioritized variants that are written to the output file in the given path. The sunburst chart shows, from the inner to the outer circle, the most severe consequence, the gene symbol, and the Ensembl transcript ID. The applied bash commands are shown below.

Moreover, we extract the SIFT (Sorting Intolerant From Tolerant) score from the Variant Effect Predictor. The SIFT Score indicates whether an amino acid substitution affects protein function<sup>8</sup>, it ranges from 0 to 1, with low scores indicating a damaging effect.

The user can now manually inspect the resulting table and decide which variants should be exported to the final result file. Using the interactive sliders, it is very easy to remove variants that are likely to have a low impact on the disease under consideration. In addition, it is possible for the domain expert to add a manual annotation that will also be written to the result file.

As can be seen in Figure 48, we have filtered for the SIFT score as well as the minor allele frequency in the 1000 genomes project and in the gnomAD data with a cut off of 0.05. This allows us to narrow down the set of potentially harming variants from variants in 45,106 transcripts to six. Those six include the one variant in the HBB gene that we purposefully introduced at the beginning. That missense mutation that we successfully found occurs in the Database [dbSNP](#), where we can learn that the homozygous variant at position chr11:5227002 (GRCh38) can lead to [Sickle cell disease](#). Hence, we were successful in interactively finding the disease-causal variant among large numbers of variants in our dataset.

Lastly, the selected results and a summary of the used commands (command line and GET Request) are displayed in the final view (Figure 49).

## Summary

We created a reproducible workflow for a typical bioinformatics application: variant prioritization. This workflow can be easily accessed via the new KNIME WebPortal to allow for domain expert interactions. In the workflow, we combined typical command line tools with built-in functionality in KNIME Analytics Platform including shared components, REST Services, and interactive visualizations. This allowed us to filter and predict the effect of thousands of variants and to find a disease-causing one within our data set.

## References

1. McLaren, W., Gil, L., Hunt, S.E. et al. The Ensembl Variant Effect Predictor. *Genome Biol* 17, 122 (2016). <https://doi.org/10.1186/s13059-016-0974-4>
2. Cooper, G., Shendure, J. Needles in stacks of needles: finding disease-causal variants in a wealth of genomic data. *Nat Rev Genet* 12, 628–640 (2011). <https://doi.org/10.1038/nrg3046>
3. Petr Danecek, Adam Auton, Goncalo Abecasis, Cornelis A. Albers, Eric Banks, Mark A. DePristo, Robert E. Handsaker, Gerton Lunter, Gabor T. Marth, Stephen T. Sherry, Gilean McVean, Richard Durbin, 1000 Genomes Project Analysis Group, The variant call format and VCFtools, *Bioinformatics*, Volume 27, Issue 15, 1 August 2011, Pages 2156–2158, <https://doi.org/10.1093/bioinformatics/btr330>
4. McLaren, W., Gil, L., Hunt, S.E. et al. The Ensembl Variant Effect Predictor. *Genome Biol* 17, 122 (2016). <https://doi.org/10.1186/s13059-016-0974-4>
5. GCCL Cardenas, Raony, et al. "Mendel, MD: a user-friendly open-source web tool for analyzing WES and WGS in the diagnosis of patients with Mendelian disorders." *PLoS computational biology* 13.6 (2017): e1005520
6. Sefid Dashti, Mahjoubeh Jalali, and Junaid Gamieldien. "A practical guide to filtering and prioritizing genetic variants." *Biotechniques* 62.1 (2017): 18-30. <https://www.future-science.com/doi/10.2144/000114492>

7. Karczewski, K.J., Francioli, L.C., Tiao, G. et al. The mutational constraint spectrum quantified from variation in 141,456 humans. *Nature* 581, 434–443 (2020). <https://doi.org/10.1038/s41586-020-2308-7>
8. Pauline C. Ng, Steven Henikoff, SIFT: predicting amino acid changes that affect protein function, *Nucleic Acids Research*, Volume 31, Issue 13, 1 July 2003, Pages 3812–3814, <https://doi.org/10.1093/nar/gkg509>

# **Chapter 3: Text Mining**

In this chapter we want to introduce three examples of using KNIME for text mining.

## **Predicting the Purpose of a Drug**

This is a quite complex story where we show how to train a named-entity (NER) recognition model that automatically detects drug names in biomedical literature. In addition, we predict the purpose of those drugs detected by the trained model. The process is divided into four different workflows. In the first workflow we collect the initial list of annotated drug names from the World Health Organization via REST API and create a corpus. In the second workflow the data within the corpus is preprocessed, and the NER model is trained and evaluated. In the third workflow we then use the model to tag the whole set of documents, thereby detecting not only known drug names, but also some that were not in our initial data. We create a co-occurrence network of drug names co-occurring in the same documents and also predict the purpose of a drug based on the node neighborhood. In the fourth and last workflow, we extract, visualize and validate interesting subgraphs. The trained model and the prediction process can now be applied to any new literature.

## **Will They Blend? KNIME meets the Semantic Web**

This semantic web story is not explicitly from the Life Science area, but nicely shows in a single workflow the basics of how to work with a Web Ontology Language (OWL) file. The OWL file we used here contains an ontology on pizza that is often used as an educational example. In the first step we are reading the OWL file using the Triple File Reader node. Based on this, a SPARQL Endpoint is created in order to allow the execution of SPARQL queries, with which we are extracting certain information. After some processing of the retrieved data we generate a view for the user to explore the extracted data. As an add-on, we also demonstrate how to integrate hyperlinks to Google searches in the Interactive Table view, so users can click on the table content and a new window with Google search results opens.

## **Exploring a Chemistry Ontology with KNIME**

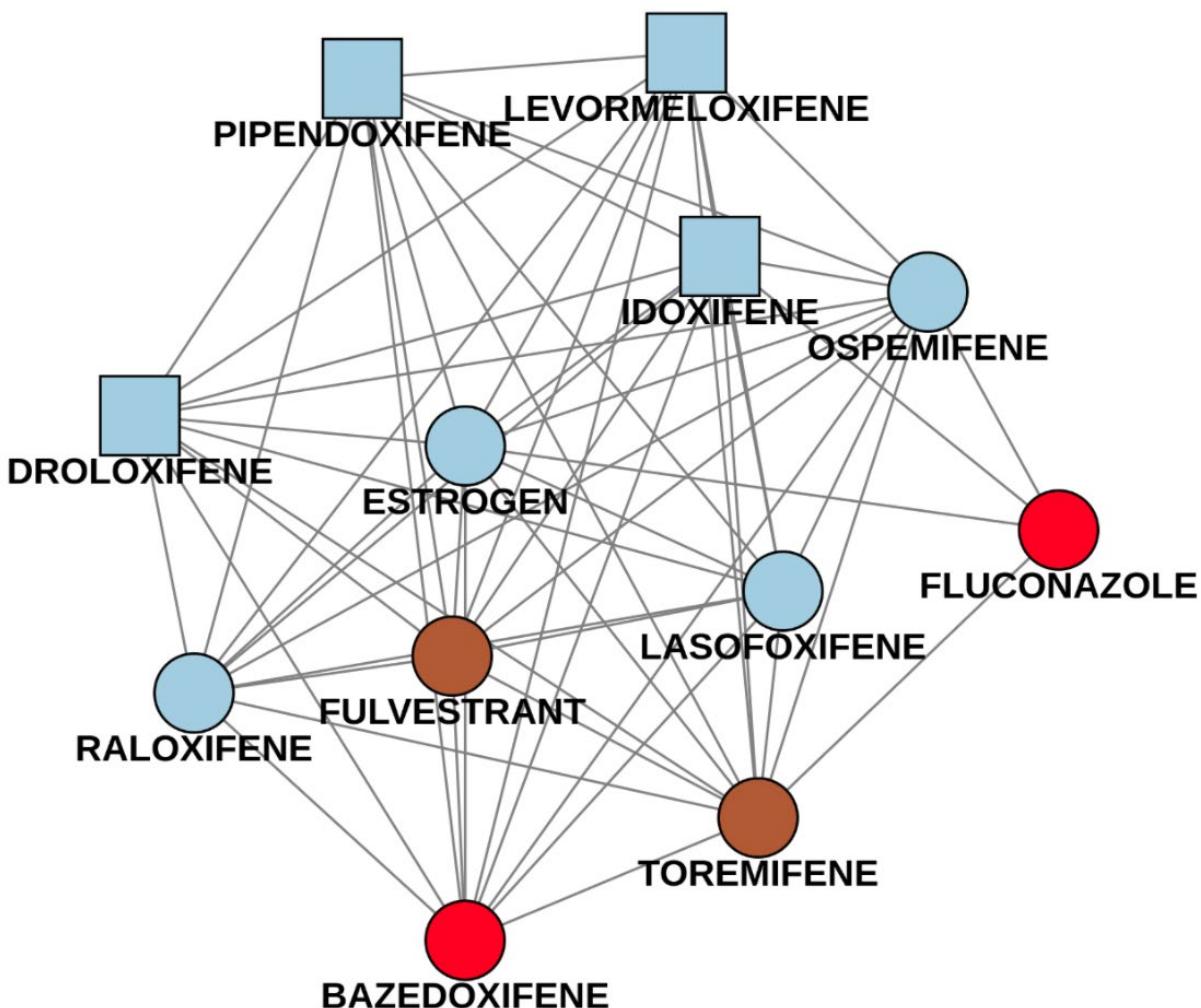
In this story we will explore another ontology, which is the ChEBI ontology that contains a classification of chemical compounds as well as information about their role. After downloading ChEBI as an OWL file, we read it into KNIME and insert the list of triples into a SPARQL Endpoint, similar to the previous story. In the next configuration step, a SMILES string for a substructure search can be entered and the biological/chemical role of the compound can be selected. All the compounds matching the substructure search and chosen role are displayed in an interactive view. After one of the compounds of interest is chosen, it is shown in two networks with all their parent classes, hierarchies and roles. The user can choose two roles here to investigate further compounds sharing these two roles in the last interactive view.

## 3.1 Predicting the Purpose of a Drug

By Julian Bunzel

Find the workflow(s) here: <https://kni.me/s/h7x1z5Px5dNWE24v>

Keeping track of the latest developments in research is becoming increasingly difficult with all the information published on the Internet. This is why Information Extraction (IE) tasks are gaining popularity in many different domains. Reading literature and retrieving information is extremely exhausting, so why not automate it? At least a bit. Using text processing approaches to retrieve information about drugs has been an important task over the last few years and is getting more and more important<sup>1</sup>.



Therefore we want to create a model that automatically detects drug names. In addition, we will go one step further and predict the purpose of those drugs detected by the trained model. This can help to get an understanding of the drugs mentioned in articles of interest and might also

help in studies about drug repurposing. Has this drug lately been mentioned together with other drugs, although they have little in common? Could there be an unknown or new connection which might help to use the drug for another purpose than usual?

Specifically, we will train a named-entity recognition (NER) model to detect drug names in biomedical literature. To do this we need a set of documents and an initial list of drug names. Since our goal is not only the recognition of these drug names but also the prediction of a drug's purpose, we need some additional information about these drugs.

After collecting the list of drug names, we will automatically extract abstracts from PubMed. These documents will be split in two parts: one part used as our training corpus to train the model and one part for testing and validation purposes. The final model is then used to tag the whole set of documents. Based on the tagged drug names, we will create a drug-drug co-occurrence network. All of our known drugs (the drugs from our initial list) will have some additional information which can be used to predict the purpose of a drug that recognized for the first time by our model (and was not in our initial list).

The work was split into four different workflows:

1. Gathering drug names and related articles
2. Preprocessing, Model Training and Evaluation
3. Create a Co-Occurrence Network and Predict Drug Purposes
4. Extract Interesting Subgraphs

## Gathering drug names and related articles

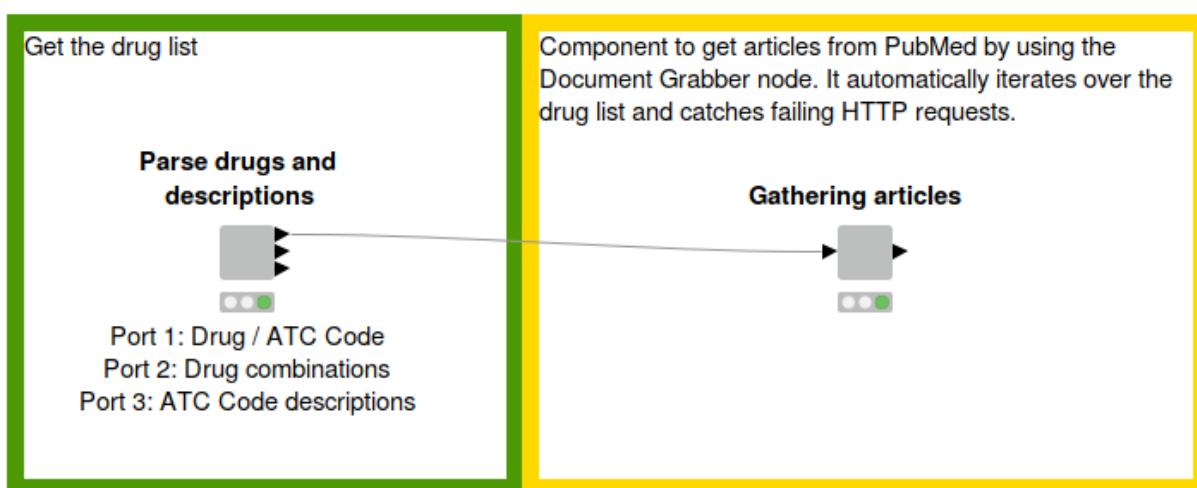


Figure 50 Workflow to parse the drug list and descriptions from the WHO website and create a corpus by using the Document Grabber to fetch articles from PubMed. The functionality is wrapped in components for better clarity.

## Dictionary creation (Drug names)

As mentioned above, our initial list of drug names should have some sort of additional information which can be used to predict the purpose of newly identified drugs. Therefore, we decided to use drugs that are covered by the Anatomical-Therapeutic-Chemical (ATC) Classification System<sup>2</sup>, which is published by the World Health Organization. It contains around 800 drug and drug combinations whereas each drug is associated to one or more ATC codes.

## ATC Classification System

[The ATC code](#) itself consists of seven letters and is separated into five different levels. As an example there is acetylsalicylic acid (aspirin) with ATC codes A01AD05, B01AC06 and N02BA01.

The first letter stands for one of fourteen anatomical main groups which will be used for ATC code prediction later. The succeeding two letters describe the therapeutic subgroup, followed by one letter describing the therapeutic/pharmacological subgroup. The fourth level is resembled by the fifth letter and stands for the chemical/therapeutic/pharmacological subgroup.

The last two digits then indicate the generic drug name. For example A01AD05, there is A for alimentary tract and metabolism, A01 for stomatological preparations, A01AD for other agents for local oral treatment and finally A01AD05 for the compound's name acetylsalicylic acid.

## Corpus creation

In the next step, we can start to gather abstracts related to the drugs from our drug list. As a source for biomedical literature we chose the widely-known PubMed database. To retrieve articles from PubMed and put them into KNIME we can use the Document Grabber node. It fetches a certain number of articles for each provided drug name in our drug list. In this case we try to get 100 articles per drug name.

## Preprocessing, model training and evaluation

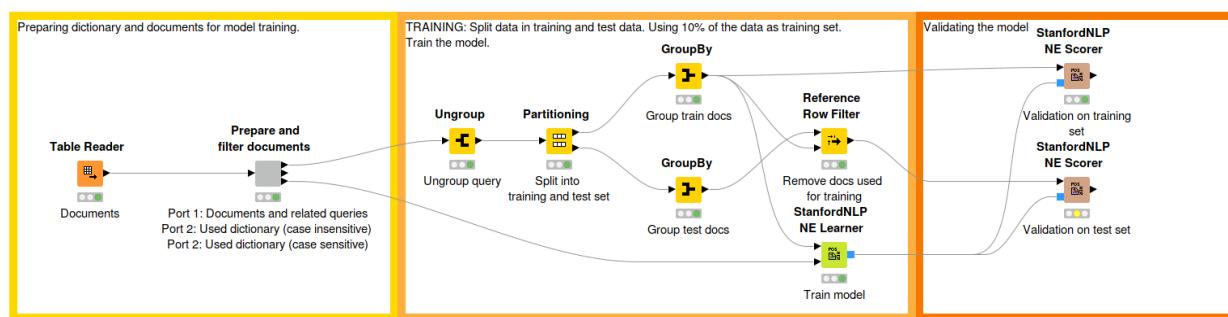


Figure 51 Workflow that describes the model training process. The first part reads the text corpus created in the first workflow and preprocesses and filters some articles. The middle part shows the model training while the third part is for evaluation.

To guarantee the quality of our corpus, some preprocessing tasks are required:

At first we check if the downloaded articles contain the query term. PubMed sometimes provides articles that are related to a drug with a similar name, but does not exactly contain the search

query. Since we can't prepare the abstracts for model training then, we filter the unrelated articles. This can be done by tagging the articles with the Wildcard Tagger node and removing an article in case no word could be tagged. Afterwards we remove all drugs from our drug list with less than 20 remaining articles to ensure a dataset with enough sample sentences containing the drug names. This yielded a final text corpus of 44891 unique abstracts (53311 with duplicates) with 207875 annotated drug named-entities in total.

Now, we can start to train our model, but before we do this, we partition our data into a training and a test set. We used 10% of the data for the training set (approx. 5000 articles). To train the model we use the StanfordNLP NE Learner node which currently provides 15 different parameters. Some important parameters are the useWord (set to true), useNGrams (set to true), maxNGramLength (set to 10) and noMidNGrams (set to true) parameters. They define whether to use the whole word, as well as substrings of the word as a feature, how long these substrings can be and whether the substrings can only be used as feature if they are taken from the beginning or end of the word. These things might not always be relevant, but in terms of drug names which often share similar word stems, it's quite useful.

Another important setting is the case sensitivity option of the learner node. Since we don't know which case is used for the words within our corpus, we choose case insensitivity, so that no matter which case is used, it is labeled by the learner node.

After training the model, we can evaluate the model by using the StanfordNLP NE Scorer. It tags the documents once by using regular expressions and once by using the trained model and compares the tags. The resulting table provides basic scores like precision, recall and counts for true and false positives/negatives.

|               | Precision | Recall | F1    | TP     | FP   | FN   |
|---------------|-----------|--------|-------|--------|------|------|
| training data | 0.996     | 0.996  | 0.996 | 26195  | 108  | 108  |
| test data     | 0.983     | 0.99   | 0.987 | 179668 | 3110 | 1773 |

Table 1 : This table shows the number of true/false positives, false negatives as well as some basic metrics like precision, recall and f1.

As we can see the majority of drug names could be tagged correctly. False positives are not necessarily a problem, because the model is not only for tagging known drug names from our initial drug list, but it also generalizes to find new entities. Otherwise we could have just used the Wildcard or Dictionary Tagger. However, regarding false positives, we still don't know if the newly tagged words are drug names at all, but we will investigate it later.

Since the Scorer node only gives us counts and scores, we use an additional approach for evaluation which helps us to identify the words causing false positives and negatives. Basically, we do what the StanfordNLP NE Scorer node does internally. We tag the documents twice, once using the StanfordNLP NE Tagger and once using the Wildcard Tagger. Afterwards we count the number of annotations for each drug and compare the different tagging approaches. For most drugs, we can see that they were annotated at the same frequency, no matter which annotation

method was used. For the example of insulin, we can see that the model sometimes just tagged insulin although there was another name component (e.g. aspart or degludec).

|                  | Annotations by regex | Annotations by model |
|------------------|----------------------|----------------------|
| INSULIN          | 455                  | 472                  |
| INSULIN ASPART   | 16                   | 11                   |
| INSULIN DEGLUDEC | 24                   | 21                   |
| INSULIN DETEMIR  | 8                    | 7                    |
| INSULIN GLARGINE | 44                   | 38                   |
| INSULIN LISPRO   | 25                   | 23                   |

Table 2 This table shows the number of annotations for each insulin related drug by using regex and the trained model. As we can see, the model annotated insulin more often than actually available in the literature since it failed to detect the second part. This helps to identify the number of false positives from Table 1.

Apart from all the measurements, we of course want to know what kind of newly identified entities there are. To get a small overview, we can use the String Matcher node to identify similarities between new words and drug names from our initial list. After doing so, we see that there are some words that are just spelling mistakes or slight variations of the drug name due to different spellings in other countries. Some newly found names were just extensions of known drugs (e.g. insulin isophane). However, in the end we were able to detect around 750 new words whereas more than half could not be linked to a drug name from the initial list. These words would need further investigation.

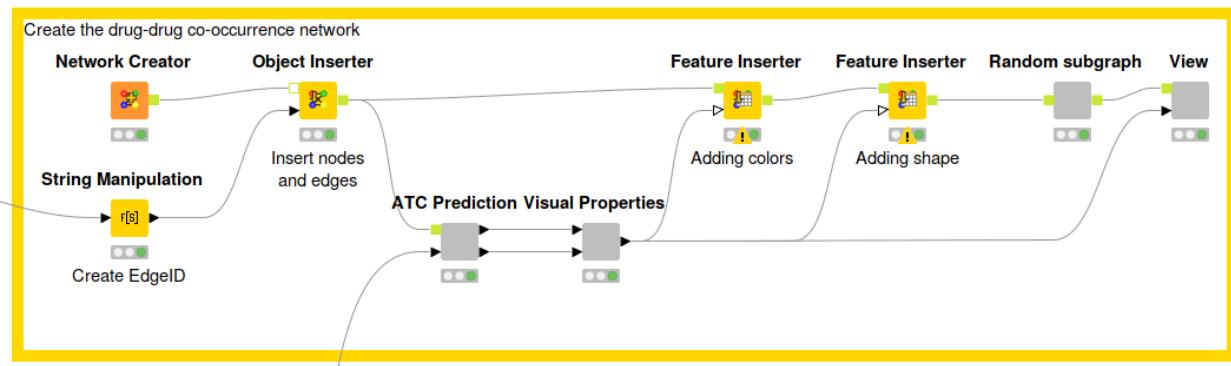


Figure 52 : Workflow that describes the network creation process. First, we use the Network Creator node to create an empty network that can be filled with new nodes and edges by using the Object Inserter. Afterwards, we predict the ATC codes and create visual properties (color & shape) for the nodes within the network. These properties can be added by using the Feature Inserter node. After doing so, we use the Network Viewer JS (hidden in the View component) to visualize the network. Download the Co-occurrence Network workflow from the KNIME Hub.

## Create a co-occurrence network and predict drug purposes

Enough about training and evaluating the model. Let's make use of the model. We can use the drug names tagged by our model to create a co-occurrence network of drug names co-occurring in the same documents. This allows us to investigate the newly found drug names in more detail and, furthermore, enables the prediction of the purpose of those newly identified drugs.. To create that network, we use the Term Co-occurrence Counter node which counts co-occurrences on

sentence or document level. In this case it is enough to set it to document level, since our documents are high-level abstracts and it's very likely that drugs being named together in an abstract are somehow related. Based on our resulting term co-occurrence table, we can create a network.

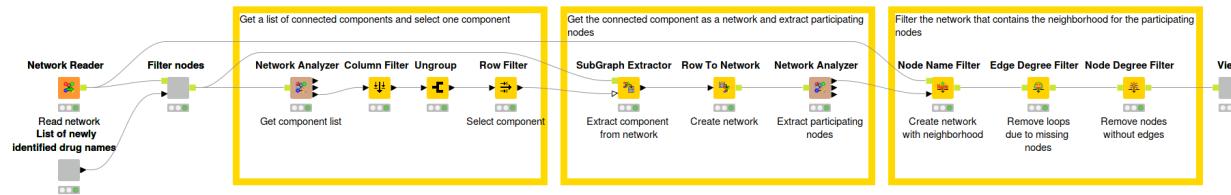


Figure 53 Workflow that describes the process of extracting small connected components to investigate the predictions.

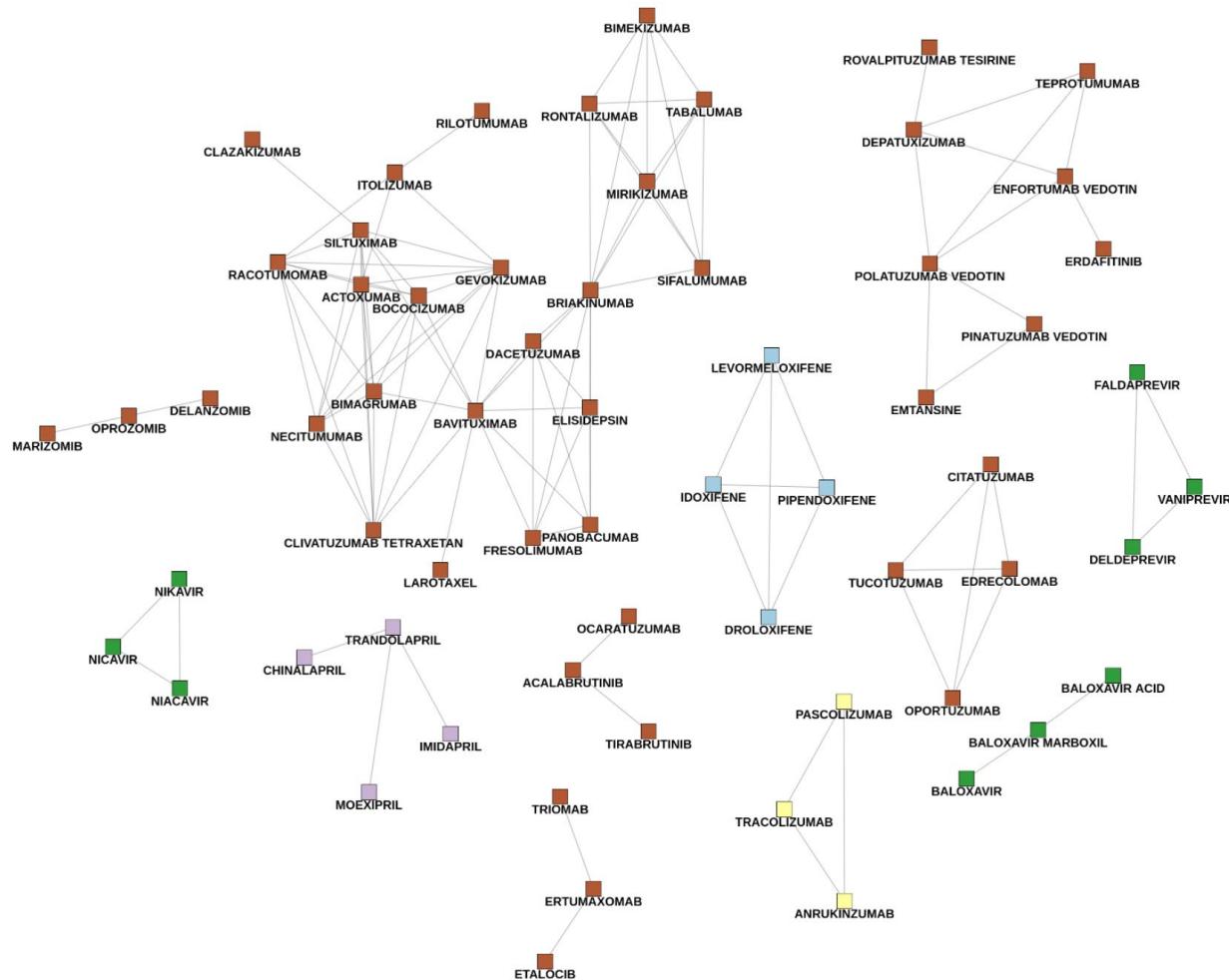


Figure 54 Connected components consisting of unknown drugs only. The color describes different types of drugs based on their predicted ATC code. Each of these components will be extended with their neighbors from the complete network to evaluate the ATC predictions.

This network can now be used to predict ATC codes of drugs. For each of our newly identified drug names, we do a majority vote, meaning that we set the ATC code that occurred most

frequently in the neighborhood of an “unknown” drug. For visualization purposes, all drugs in the network are colored based on the first level of the ATC code. Additionally, newly detected drugs are displayed as squares and known drugs as circles, respectively. This helps to evaluate and comprehend the prediction of the ATC code. As mentioned before, our initial list had around 800 drug names and the list of newly found entities contains 750 drugs. So in total there are quite some nodes in the network which makes the view pretty confusing. To avoid this, I show you how to extract relevant subgraphs to evaluate and comprehend predictions in the next section.

## Extract interesting subgraphs

To investigate the predictions in detail, we can use connected components of newly detected drugs. At first, we remove all drugs from the network that are in the initial drug list to get a set of nodes in the network containing only the newly identified drugs. Afterwards, we re-add all of the previously filtered drugs that are in the first neighborhood of drugs from the component we are looking at. In the end, each connected component consists of a set of co-occurring newly detected drug names plus their neighbors from the initial network. This approach makes evaluating easier since we first filter mostly drugs from the initial drug list that tend to be connected to a huge number of drugs, but later re-add these to a smaller set of unknown drugs.

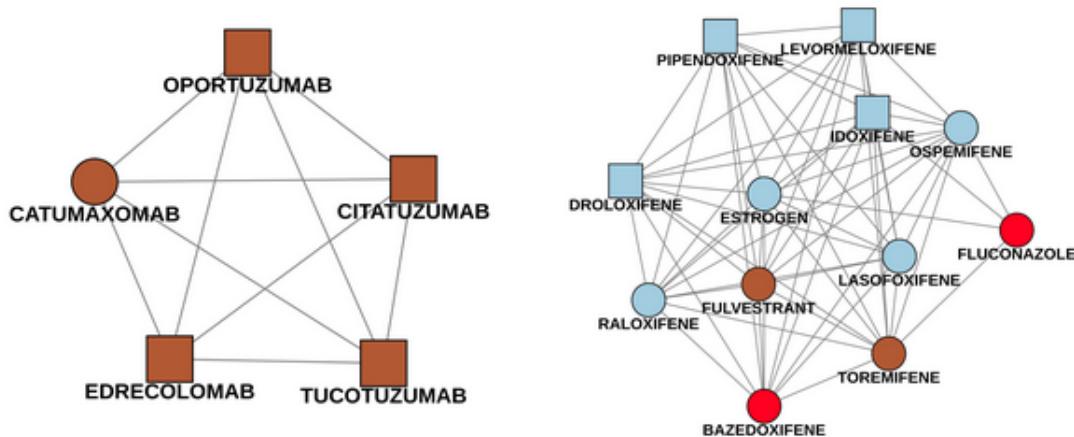


Figure 55 Two example subgraphs.

Our example (Figure 55) shows two of these subgraphs. The first picture is an easy case, since the four newly identified drug names only have one connection to a known drug (catumaxomab). All drugs were labeled as Antineoplastic and immunomodulating agents which is indeed correct. The second component is trickier. There are four newly detected drugs pipendoxifene, levormeloxifene, idoxifene and droloxifene. All of them were predicted as Genito-urinary system and sex hormones, since most of the known drugs in the network are in this ATC class (bazedoxifene included - it's colored red because it has multiple ATC classes). However, there are also connections to Antineoplastic and immunomodulating agents like fulvestrant and toremifene. Connections to both of these drugs are worth mentioning as well, since the new drugs were mostly developed for breast cancer treatments. As we can see, the prediction might be right,

but having a look at connections to ATC classes with a lower influence is also helpful to understand the purpose in a better way.

## **Summary**

We successfully trained a named-entity recognition model to detect drug names in biomedical literature and predict the purpose of the newly identified drugs. We started with an initial set of drug names from the World Health Organization, which also provides some more information about the drug's purpose as they are annotated using the ATC Classification System. Based on this list, we then created a text corpus of articles by fetching them from PubMed. The StanfordNLP NE nodes then helped to train a named-entity recognition model to detect not only known drug names, but also some that were not in our initial data. Finally, we built a drug co-occurrence network to predict the purpose of unknown drugs based on their neighborhood and showed how to extract interesting subgraphs to easily evaluate our predictions.

The trained model and the prediction process can now be applied to any new literature, to get an instant overview of all drugs mentioned.

## **References**

1. "Drug name recognition and classification in biomedical texts. A case ...."17 July 2008. Accessed 12 September 2019
2. "ATC/DDD Index - WHOCC"13 December 2018. Accessed 12 September 2019

## 3.2 Will They Blend? KNIME meets the Semantic Web

By Martyna Pawletta

Find the workflow(s) here: <https://kni.me/w/XYhcwbBj7In9hb70>

Ontologies – or let's see if we can serve pizza via the semantic web and KNIME Analytics Platform. Will they blend?

Ontologies study concepts that directly relate to “being” i.e. concepts that relate to existence and reality as well as the basic categories of being and their relations. In information science, an ontology is a formal description of knowledge as a set of concepts within a domain. In an ontology we have to specify the different objects and classes and the relations - or links - between them. Ultimately, an ontology is a reusable knowledge representation that can be shared. Check out the [Linked Open Data Cloud](#) which has an amazing graphic showing how many ontologies (linked data) there are available in the web.

### The Challenge

The Semantic Web and the collection of related Semantic Web technologies like [RDF](#) (Resource Description Framework), [OWL](#) (Web Ontology Language) or [SPARQL](#) (SPARQL Protocol and RDF Query Language) offer a bunch of tools where linked data can be queried, shared and reused across applications and communities. A key role in this area is played by ontologies and OWLs.

So where does the OWL come into this? Well, no - we do not mean the owl as a bird here – but you see the need for ontologies, right? An OWL can have different meanings, and this is one of the reasons why creating ontologies for specific domains might make sense.

Ontologies can be very domain specific and not everybody is an expert in every domain - but it's a relatively safe bet to say that we've all eaten pizzas at some point in time - so let's call ourselves pizza experts. Today's challenge is to extract information from an OWL file containing information about pizza and traditional pizza toppings, store this information in a local SPARQL Endpoint, and execute SPARQL queries to extract some yummy pizza, erm - I mean data. Finally, this data will be displayed in an interactive view which allows you to investigate the content.

The ontology used in this blog post and demonstrated workflow is an example ontology that has been used in different versions of the Pizza Tutorial run by Manchester University. See more information on [Github](#).

## The Experiment

Reading and querying an OWL file

In the first step the Triple File Reader node extracts the content of the pizza ontology in the OWL file format and reads all triples into a Data Table. Triples are a collection of three columns containing a subject(URI), a predicate(URI) and an object(URI or literal), short: sub, pred, obj. The predicate denotes relationships between the subject and the object. As shown in the screenshot below (Figure 56), in the example we see that the Pizza FruttiDiMare is a subClassOf the class NamedPizza and has two labels: a preferred and an alternative one.

| Row ID | \$ sub  | \$ pred   | \$ obj  |
|--------|---|---|---|
| Row110 | <http://www.co-ode.org/ontologies/pizza/pizza.owl#FruttiDiMare> | <http://www.w3.org/2000/01/rdf-schema#subClassOf> | <http://www.co-ode.org/ontologies/pizza/pizza.owl#NamedP> |
| Row111 | <http://www.co-ode.org/ontologies/pizza/pizza.owl#FruttiDiMare> | <http://www.w3.org/2004/02/skos/core#prefLabel>   | 'Frutti Di Mare'@en                                       |
| Row112 | <http://www.co-ode.org/ontologies/pizza/pizza.owl#FruttiDiMare> | <http://www.w3.org/2004/02/skos/core#altLabel>    | 'Frutti Di Mare Pizza'@en                                 |

Figure 56 Screenshot showing the output of the Triple File Reader node containing a subject, predicate and object column.

Once the Triple File Reader is executed, a SPARQL Endpoint can be created using the Memory Endpoint together with the SPARQL Insert node. This allows the execution of SPARQL queries. Note that our Triple File Reader does not officially support the OWL format. KNIME can read RDF files and consequently because OWL files are very similar we can read these files too. However not all information is necessarily retrieved as OWL can have additional parameters.

The example in Figure 57 shows a SPARQL query node that contains a query to extract a basic list with all pizzas included in the owl file.

A recommendation here: if the ontology you want to query is new to you – I would highly recommend exploring the structure and classes first quickly in another tool like [Protégé](#). This makes it easier later to create and write SPARQL queries.

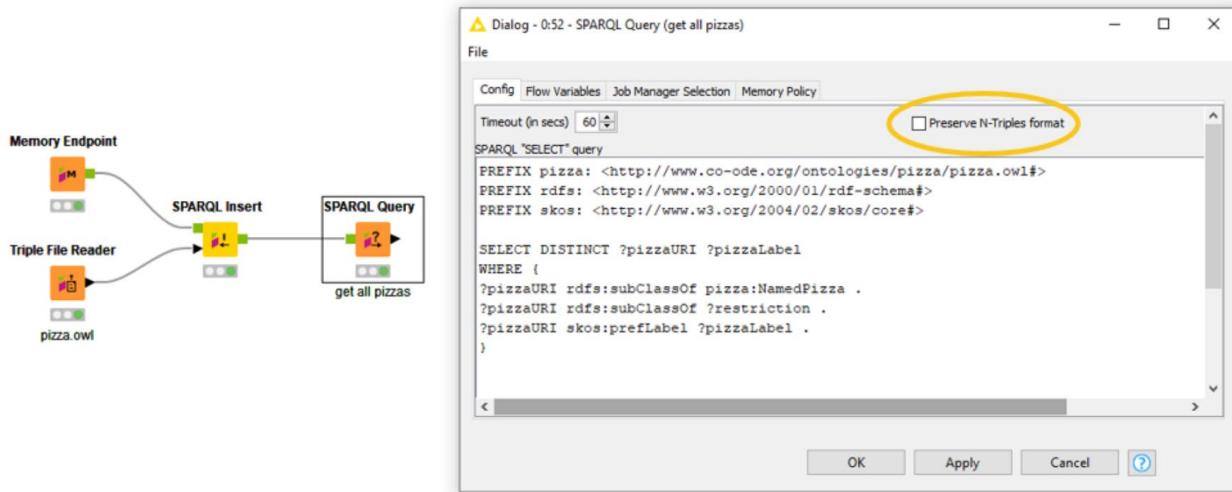


Figure 57 Example workflow that shows how to read an OWL file, insert extracted triples into a SPARQL endpoint and execute a SPARQL query to extract all kinds of pizzas from the pizza ontology.

The SPARQL query node has a checkbox on the top right (see Figure 57) saying “Preserve N-Triples format”. Selecting this makes a difference in terms of what the output data will look like. The N-Triples format needs to be kept if the triples will be inserted into an endstore.

The example below shows the effect of not checking (top) or checking (bottom) the N-triples checkbox. In case of URIs the angled brackets are not preserved, in terms of literals quotes and type (here @en) will be removed if nothing has been selected.



**Query Result Table - 0:52 - SPARQL Query (get all pizzas)**

File Hilite Navigation View

Table "default" - Rows: 22 Spec - Columns: 2 Properties Flow Variables

| Row ID | S pizzaURI  | S pizzaLabel    |
|--------|---|-----------------|
| Row0   | http://www.co-ode.org/ontologies/pizza/pizza.owl#Veneziana      | Veneziana       |
| Row1   | http://www.co-ode.org/ontologies/pizza/pizza.owl#SloppyGiuseppe | Sloppy Giuseppe |
| Row2   | http://www.co-ode.org/ontologies/pizza/pizza.owl#Soho           | Soho            |
| Row3   | http://www.co-ode.org/ontologies/pizza/pizza.owl#LaReine        | La Reine        |
| Row4   | http://www.co-ode.org/ontologies/pizza/pizza.owl#Napoletana     | Napoletana      |
| Row5   | http://www.co-ode.org/ontologies/pizza/pizza.owl#FruttiDiMare   | Frutti Di Mare  |

**Query Result Table - 0:52 - SPARQL Query (get all pizzas)**

File Hilite Navigation View

Table "default" - Rows: 22 Spec - Columns: 2 Properties Flow Variables

| Row ID | S pizzaURI  | S pizzaLabel         |
|--------|---|----------------------|
| Row0   | <http://www.co-ode.org/ontologies/pizza/pizza.owl#Veneziana>      | "Veneziana"@en       |
| Row1   | <http://www.co-ode.org/ontologies/pizza/pizza.owl#SloppyGiuseppe> | "Sloppy Giuseppe"@en |
| Row2   | <http://www.co-ode.org/ontologies/pizza/pizza.owl#Soho>           | "Soho"@en            |
| Row3   | <http://www.co-ode.org/ontologies/pizza/pizza.owl#LaReine>        | "La Reine"@en        |
| Row4   | <http://www.co-ode.org/ontologies/pizza/pizza.owl#Napoletana>     | "Napoletana"@en      |
| Row5   | <http://www.co-ode.org/ontologies/pizza/pizza.owl#FruttiDiMare>   | "Frutti Di Mare"@en  |

Figure 58 Examples of the effect of not checking (top) or checking (bottom) the N triples checkbox.

## Visualization

There are different ways in KNIME to visualize data. In the case of ontologies it's really depending on what you are aiming to do. Here we will extract a bit more information than in the first example and create an interactive view within a component that allows us to explore the content of the pizza ontology.

Additionally to the pizza labels now using two SPARQL query nodes (see Figure 59), further information like toppings per pizza type or its spiciness was extracted. Also, we query for pizza toppings that are a subclass of the class VegetableToppings and create a flag if the topping is a vegetable or not using the Constant Value Column node.

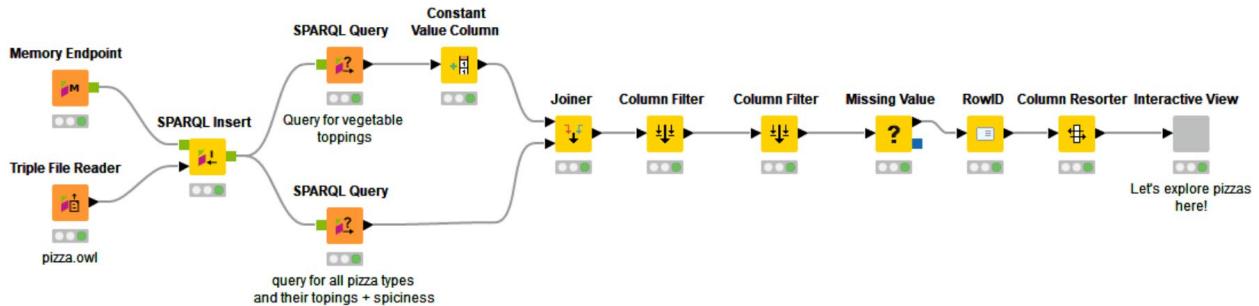


Figure 59 Example workflow showing how the basic example from Figure 57. Can be extended and an interactive view created.

Finally we create an interactive view where the extracted data can be explored (see Figure 60). To open the interactive view, right click the “Interactive View” Component + select Interactive View.



Figure 60 Interactive view showing extracted data.

## Is it real?!

When I first looked at the dataset using the view I saw the “Sloppy Giuseppe” Pizza and directly had to google it as it was something completely new to me. I saw the toppings but was wondering if this is something really Italian? This brought me to the idea of adding another feature here in addition to the tables and charts.

If you now click on the Pizza name, a new window will open showing Google search results for that specific pizza type. I did this using the String Manipulation node, which creates a link. To make sure the link opens in a new window and not in your current view the “target=\_blank” option needs to be included.

```
Expression
1 string("<a href='https://google.com/search?q="+$pizzaLabel$+"+Pizza'"+ "target=_blank">"+$pizzaLabel$+"</a>")
```

Figure 61 Include „target=\_blank“ to open link in a new window.

## The Results

We showed today how to extract data from an OWL file, create a SPARQL Endpoint and SPARQL query. Finally we generated a view where the content can be explored.

After playing with such yummy data... hungry now? Let's order a pizza then ?

## 3.3 Exploring a Chemistry Ontology with KNIME

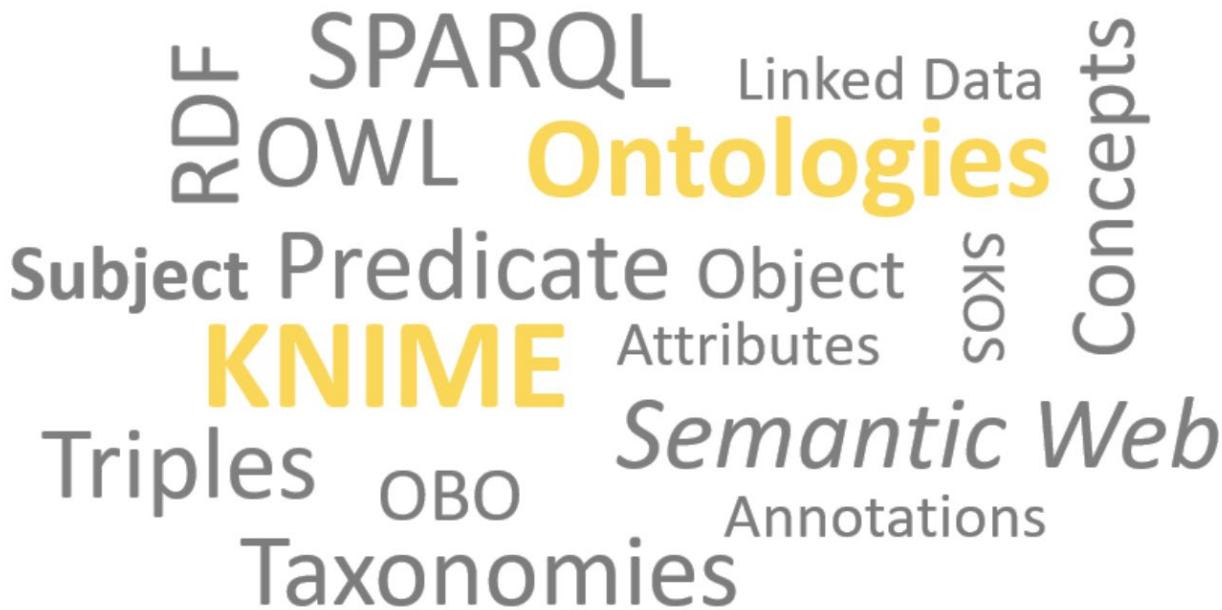
By Martyna Pawletta

Find the workflow(s) here: <https://kni.me/w/F0erB7Sb3MmoGxDc>

We are often asked if it's possible to work with ontologies in KNIME Analytics Platform.

With "work with ontologies" people can mean many different things but let's focus today on one particular ontology and basic tasks including reading and querying ontologies to create an interactive tool at the end. For this purpose, today, we dive into the world of chemistry to use the ChEBI ontology (Chemical Entities of Biological Interest).

Even if chemistry is not a domain of interest for you, this blog post can still be of high value as we, for example, show how to read an OWL file, how to create queries in SPARQL as well as different possibilities for visualizing ontology content in an interactive composite view. How you adopt this to your own use case, ontology, and extracted dataset, we will leave to you and your imagination.



### ChEBI

Especially in the Life Sciences area, ontologies are very popular and frequently used for different purposes such as data integration, curation, defining standards, or labeling. Just how important ontologies are is illustrated by facts that sources like the BioPortal 1 contain already more than 800 ontologies in their repositories.

With the workflow described in this blog post, we will demonstrate a way to explore [ChEBI](#), which is a freely available ontology containing a classification of chemical compounds as well as

information about the role of compounds like their application or the biological and chemical role. It contains in general three main classifications like chemical entity, role, and subatomic particle. In this workflow, we use molecular entity and role class (Figure 62). ChEBI can be downloaded in different file formats - today we will work with an OWL file.

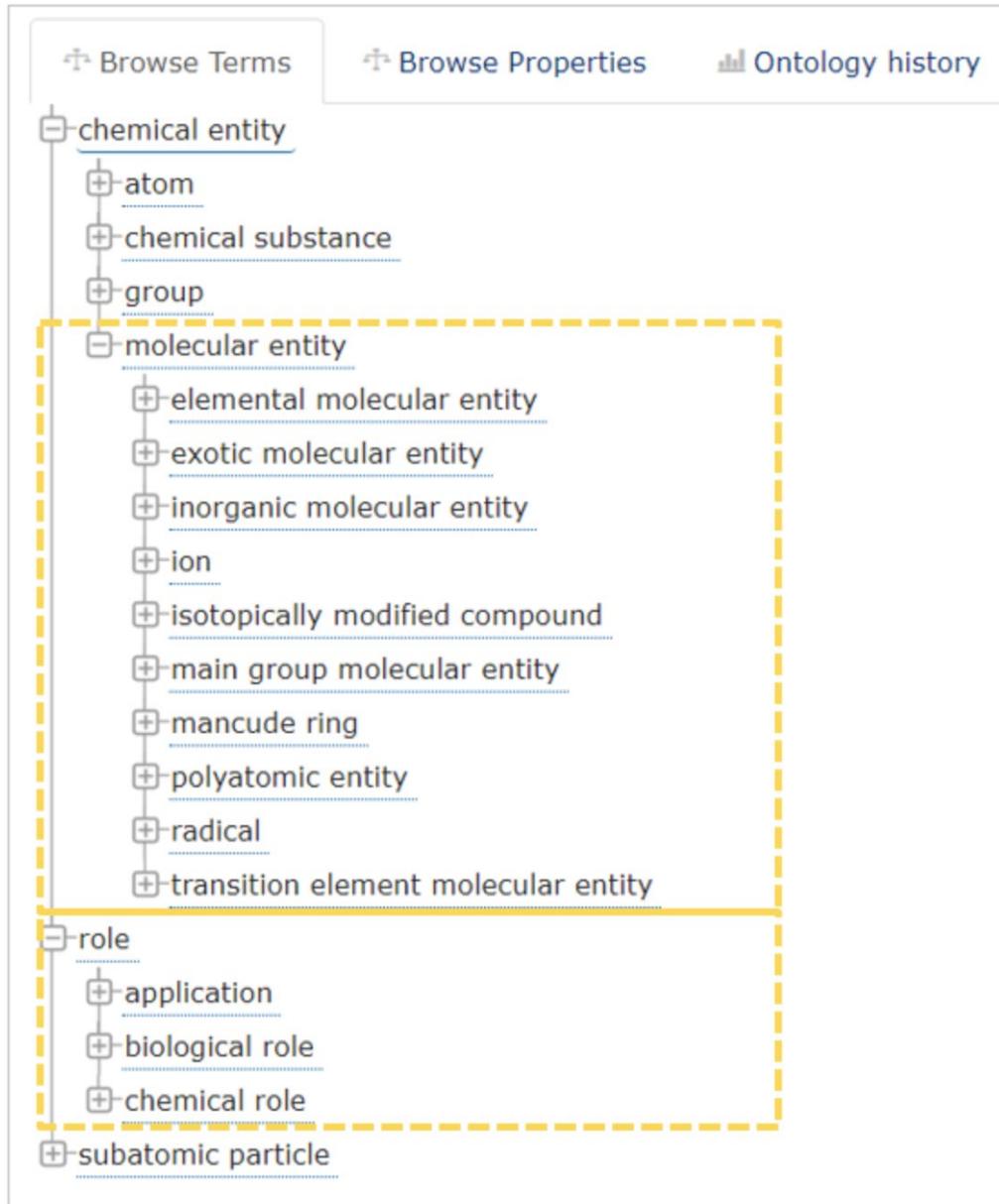


Figure 62 : Overview of ChEBI classes (screenshot taken from here). The yellow boxes show which parts will be used and explored in the workflow. The remaining parts are ignored.

## Let's start!

How to basically read and query ontologies stored in the OWL format is described in the previous chapter. In the example described in this article, we used a pizza ontology to show how easy it is to explore that type of data.

With the following example workflow, we play with the terms and content of the ChEBI ontology while combining searches, results and data in order to create interactive views where the content can be explored. We hope to learn about compounds, their biological and chemical roles, as well as definitions and other sources that contain references to a particular compound.

This analysis was realized in the workflow depicted in Figure 63 and contains the following main steps:

**Step 1:** Read the OWL file into a SPARQL Endpoint

**Step 2:** Substructure search & selecting a role of a chemical compound

**Step 3:** View compounds matching the substructure search and role. Here one compound needs to be selected

**Step 4:** Show the selected compound in a network with all their parent classes, hierarchies and roles. Select a disease in the Tag Cloud to merge some more data in the next step

**Step 5:** Viewing results from selection in Step 4

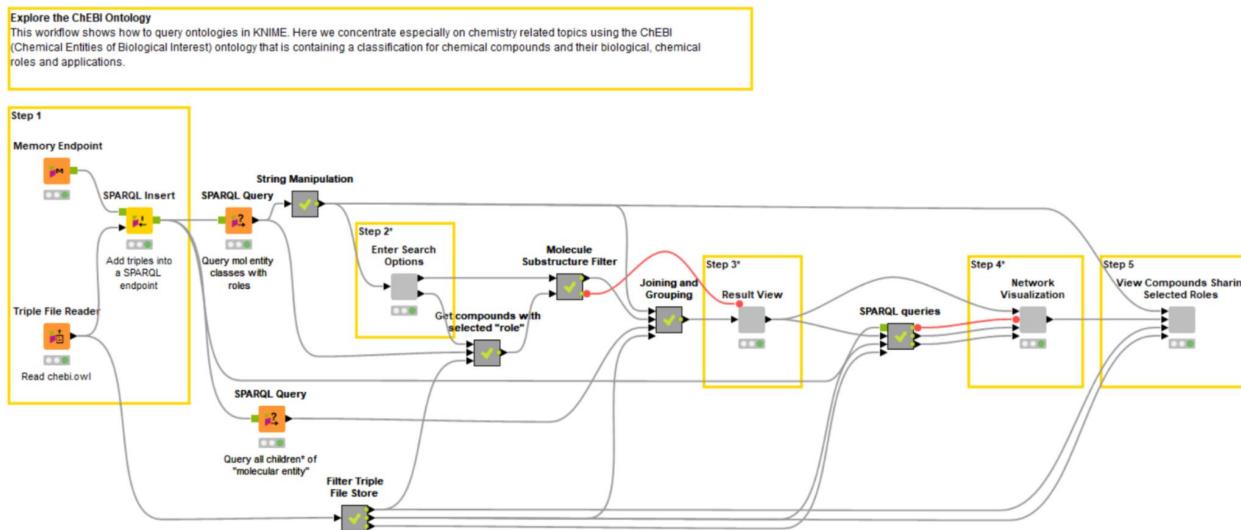


Figure 63 Example workflow showing how to explore the ChEBI ontology stored in OWL format.

### Step 1. Reading the OWL file into a SPARQL Endpoint

Analogous to the previously described use case of a pizza ontology, we use the Triple File Reader node to read the OWL file and insert the list of triples into a SPARQL Endpoint which is connected to a Memory Endpoint Node (See Figure 63, Step 1). With this in place, and successfully executed, we now have the basis to start writing and executing SPARQL queries as well as filtering information from the list of Triples.

Quick reminder here: RDF triple - also known as semantic triple - always contains three columns: subject, predicate, and object. Read more [here](#).

▲ Triples - 2:104 - Triple File Reader (Read chebi.owl)

File Help Navigation View

Table "default" - Rows: 4441655 Spec - Columns: 3 Properties Flow Variables

| Row ID     | S   sub                                     | S   pred   | S   obj  |
|------------|---|--|--|
| Row3394593 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://purl.obolibrary.org/obo/chebi/mass>                      | "284.42018"  |
| Row3394594 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://purl.obolibrary.org/obo/chebi/monoisotopicmass>          | "284.135"  |
| Row3394595 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>                | <http://www.w3.org/2002/07/owl#Class>  |
| Row3394596 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://purl.obolibrary.org/obo/IAO_0000115>                     | "A phenothiazine derivative in which the phenothiazine tricycle has a 3-(dimethylaminopropyl) group" |
| Row3394597 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://www.geneontology.org/formats/oboInOwl#hasRelatedSynonym> | "10-(3-(Dimethylamino)propyl)phenothiazine"  |
| Row3394598 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://www.w3.org/2000/01/rdf-schema#label>                     | "promazine"  |
| Row3394599 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://www.geneontology.org/formats/oboInOwl#hasRelatedSynonym> | "promazine"  |
| Row3394600 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://purl.obolibrary.org/obo/chebi/smiles>                    | "CN(C)CCN1c2ccccc2Sc2cccc12"   |
| Row3394601 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://www.geneontology.org/formats/oboInOwl#hasRelatedSynonym> | "promazine"  |
| Row3394602 | <http://purl.obolibrary.org/obo/CHEBI_8459> | <http://www.geneontology.org/formats/oboInOwl#hasRelatedSynonym> | "promazine"  |

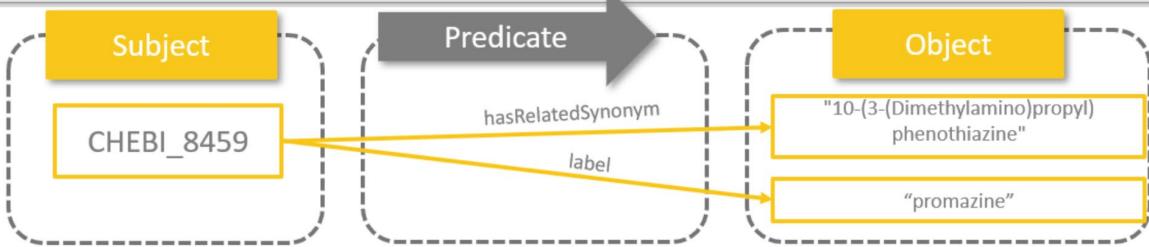


Figure 64 Schema showing how to interpret RDF triples.

## Step 2. Substructure search paired with role information

Imagine a scientist is investigating a new compound in development. She knows the chemical structure and the application of that compound but is curious to see if there are other compounds in ChEBI with similar properties. In this example workflow a SMILES for a substructure search can be added and the application or biological/chemical role of the compound can be selected (Figure 65).

Therefore, the “Enter Search Options” component will be used to create a search query in order to add the above mentioned properties (Right click the component → select interactive view).

To allow insertion of a SMILES, we have used the String Widget node. Above, in Figure 64., we added a Phenothiazine.

Let's select a dopaminergic antagonist as the role. This is frequently used in antipsychotic drugs for treating schizophrenia, bipolar disorders or psychosis stimulants.

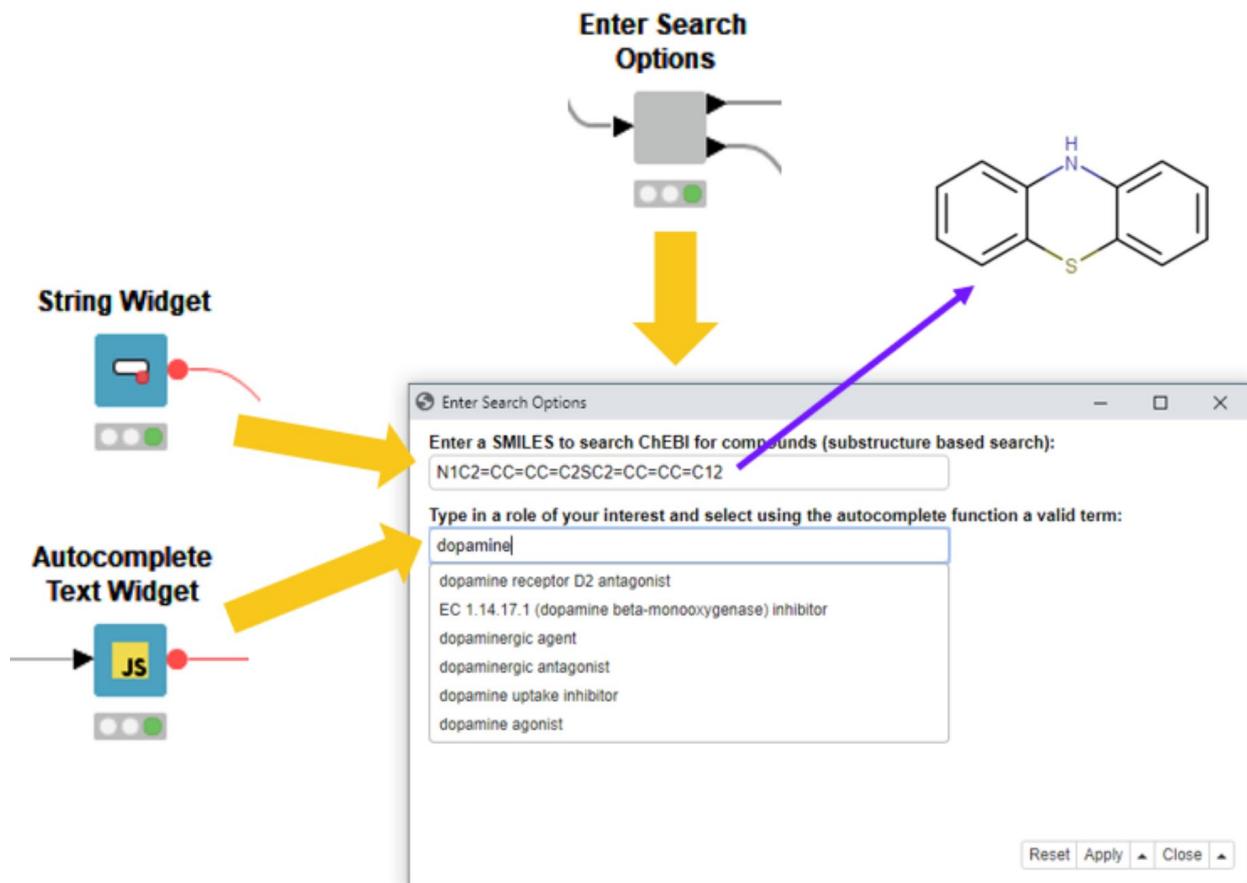


Figure 65 The “Enter Search Options” component contains two options to enter input for a substructure search as well as a search for compounds; a specific role is assigned in the ontology.

Little hint here: As an alternative to the String Widget, a Molecule String Input node could also be used. This would give you the opportunity to draw a chemical structure instead of pasting a SMILES string.

### Step 3. Let's view first results!

In the following view (“Result View” component in the example workflow) you can inspect the results of the substructure search. Here the Tile View and RDKit Highlighting were used to show all compounds matching the entered search options. All these compounds are dopaminergic antagonists as selected in the previous view.

In this example we selected a compound called Promazine to see more information.

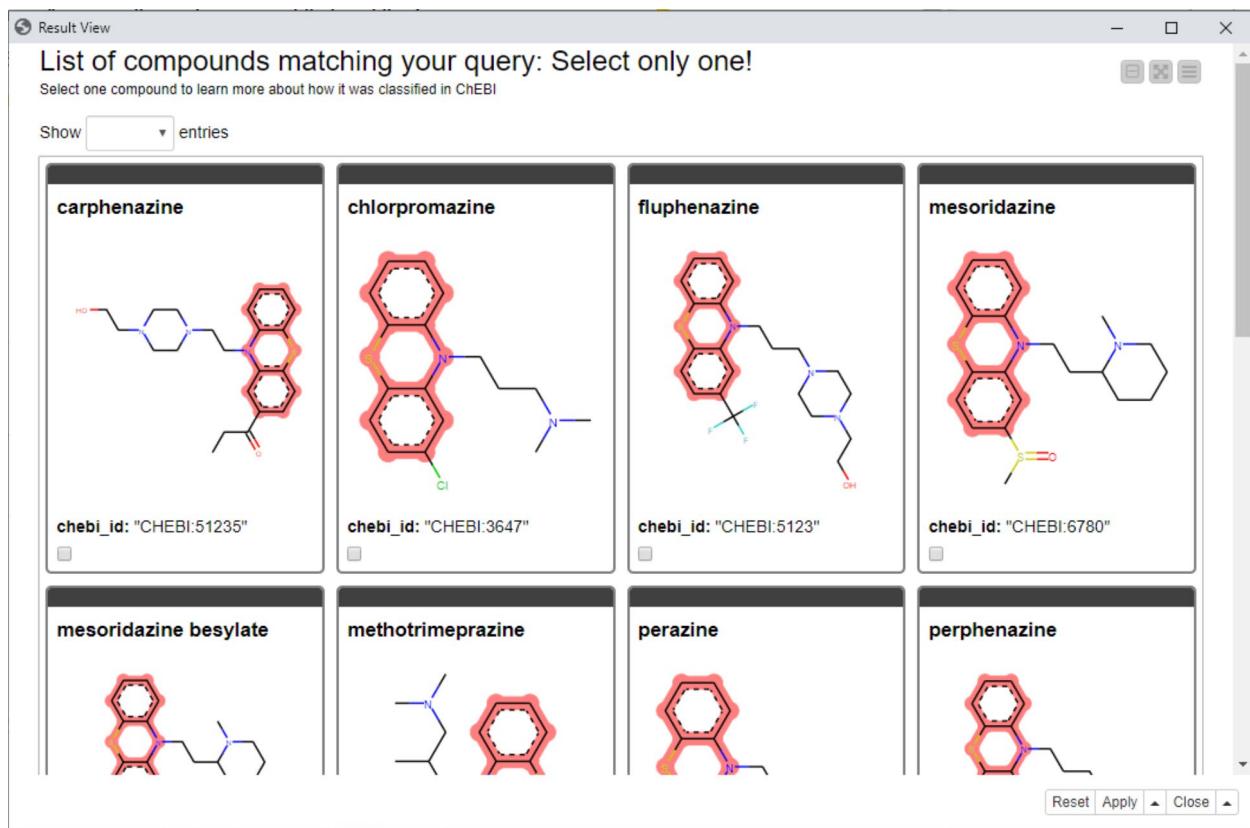


Figure 66 Result view showing chemical compounds with the highlighted substructure using the Tile View node.

#### Step 4. Show a network with class information

In this example workflow, we also would like to show a way to visualize an ontology as a network. We do this with the Network Mining Extension. The view in this step is also interactive, which means whenever a node in the network is selected, the table under the network will show more information for the extracted entity, such as the definition, for example.

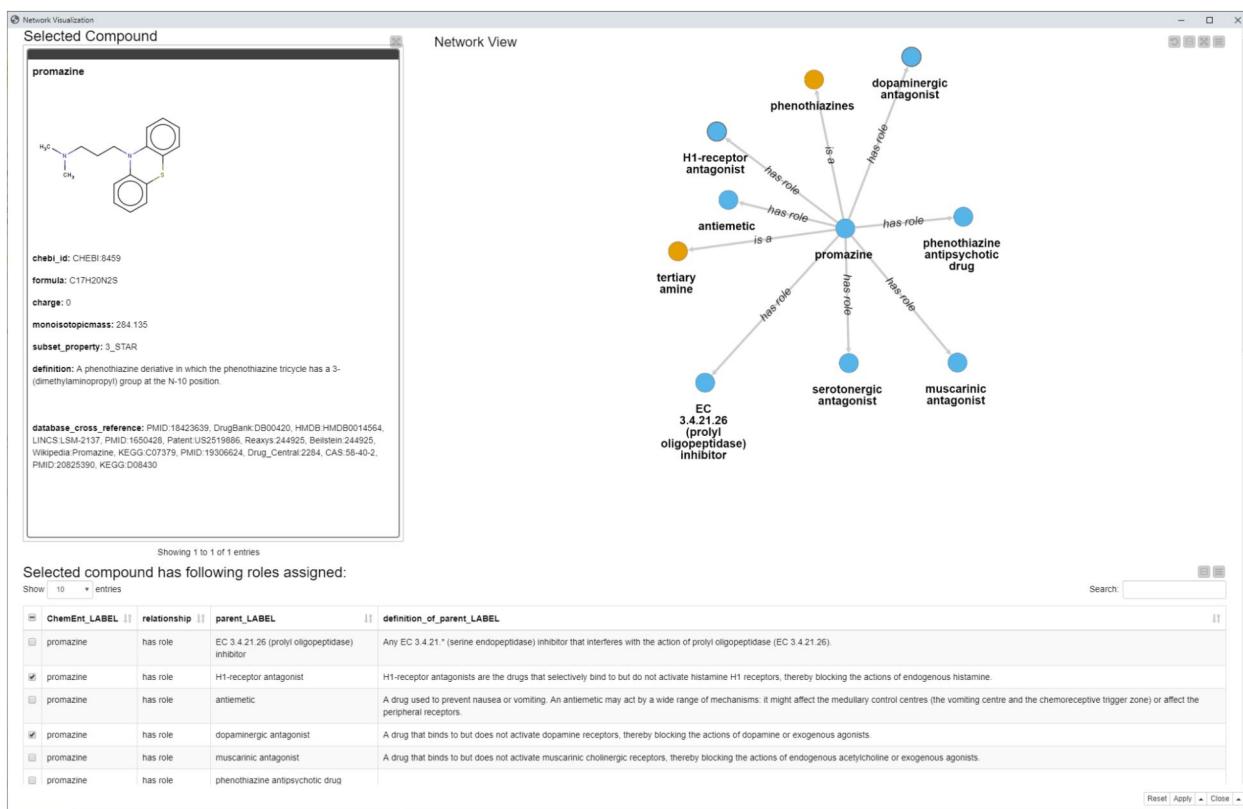


Figure 67 Network view showing the selected compound as well as a network including the subClassOf connections as “has role” and “is a”. Additionally, more information such as definitions and synonyms can be selected and made visible for a node in the network.

If you scroll down, a second network becomes visible. This network contains entities starting from the selected compound, here Promazine, and shows how it has been classified in ChEBI. It shows all “is a” links from the compound through to the chemical entity. Additionally the “has role” links were added to show also here which roles are linked (blue nodes).

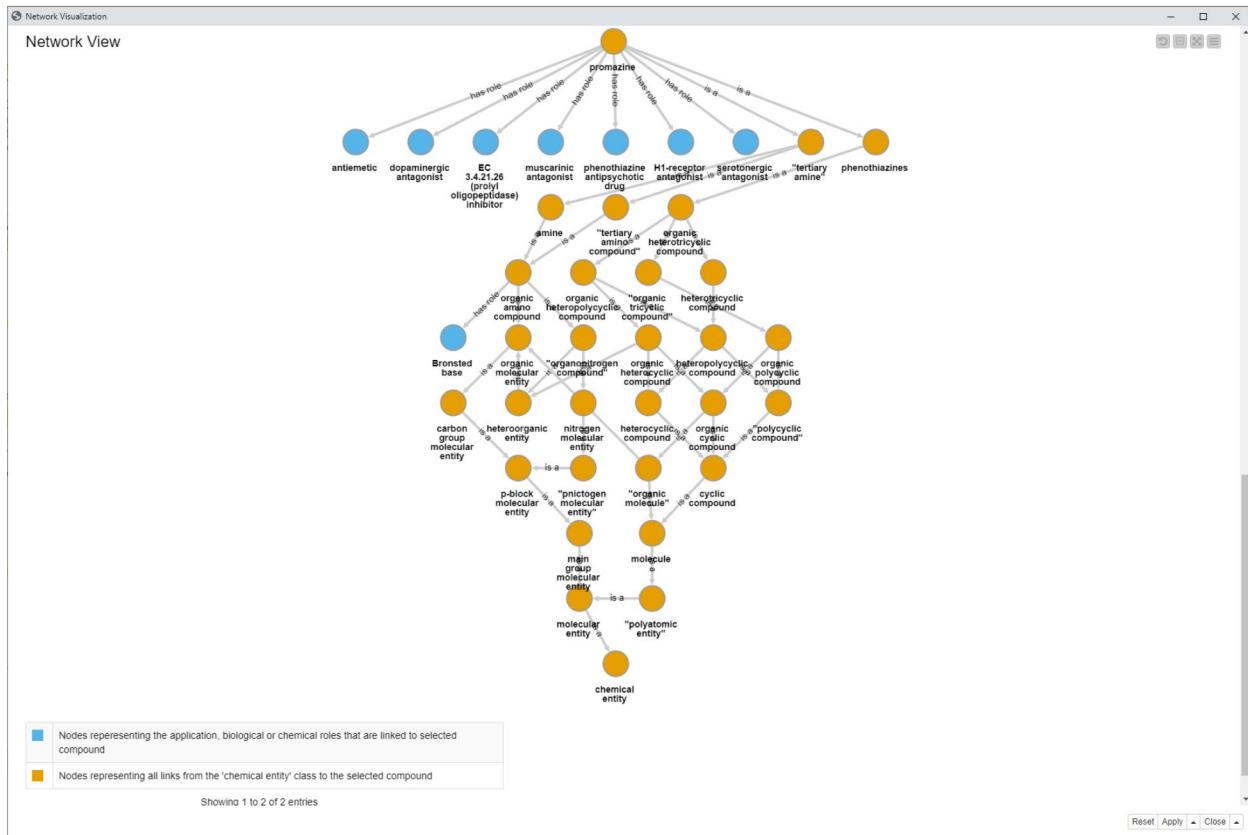


Figure 68 Network view showing the selected compound as well as a network including the subClassOf connections as “has role” and “is a”.

### Step 5. Show compounds sharing two roles

By looking at the network view in Figure 68 and investigating the roles of a compound - let's say you can now see another interesting role and you would like to see more compounds with that role - you can go one step further and select two different roles from the table like for example the already known dopaminergic antagonist in combination with the H1-receptor antagonist that plays a role in relieving allergic reactions.

In the last component “View Compounds Sharing Selected Roles” (Step 5) we see all those compounds containing both selected roles and the additional information, such as definitions, synonyms, or references to other ontologies, databases, and sources.

View Compounds Sharing Selected Roles

Show 1 entries

|                           |                           |                           |                           |                           |                           |
|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|---------------------------|
| <b>zuclopentixol</b>      | <b>clopentixol</b>        | <b>pimozide</b>           | <b>promazine</b>          | <b>risperidone</b>        | <b>thioridazine</b>       |
|                           |                           |                           |                           |                           |                           |
| formula: C22H25ClN2OS     | formula: C22H25ClN2OS     | formula: C26H29F2N3O      | formula: C17H20N2S        | formula: C23H27FN4O2      | formula: C21H26N2S2       |
| charge: 0                 |
| monoisotopicmass: 400.138 | monoisotopicmass: 400.138 | monoisotopicmass: 461.228 | monoisotopicmass: 284.135 | monoisotopicmass: 410.212 | monoisotopicmass: 370.154 |

Showing 1 to 6 of 6 entries

Previous 1 Next

Show 10 entries

Search:

| preferred_label | Role(s)   | definition   | has_exact_synonym  | has_related_synonym  | subset_property | database_cross_reference   |
|-----------------|---|--|--|--|-----------------|--|
| zuclopentixol   | alpha-adrenergic antagonist, H1-receptor antagonist, first generation antipsychotic, dopaminergic antagonist, serotonergic antagonist | "The (Z)-isomer of clopentixol."   | 2-[4-(3Z)-3-(2-chloro-10H-dibenzo[b,e]thiopyran-10-ylidene)propyl]piperazin-1-yl)ethanol | zuclopentixolum, zuclopentixol, (Z)-4-(3-(2-Chlorothioxanthen-9-ylidene)propyl)-1-piperazineethanol, zuclopentixol   | 3_STAR          | Beilstein:8447014, DrugBank:DB01624, Drug_Central:2877, CAS:53772-83-1, KEGG:D03556  |
| clopentixol     | alpha-adrenergic antagonist, serotonergic antagonist, first generation antipsychotic, dopaminergic antagonist, H1-receptor antagonist | "A thioxanthene derivative having a chloro substituent at the 2-position and an alkylidene group at the 10-position with undefined double bond stereochemistry." | 2-[4-(3-(2-chloro-9H-thioxanthen-9-ylidene)propyl)piperazin-1-ol]                        | clopentixol, 4-(3-(2-Chlorothioxanthen-9-ylidene)propyl)-1-piperazineethanol, Chloropentixol, clopentixolum, 4-(3-(2-Chloro-9H-thioxanthen-9-ylidene)propyl)-1-piperazineethanol, 2-[4-(3-(2-chloro-10H-dibenzo[b,e]thiopyran-10-ylidene)propyl)piperazin-1-yl]ethanol | 3_STAR          | Beilstein:899403, KEGG:D02613, Patent:BE585338, LINCS:LSM-2631, Drug_Central:4397, Wikipedia:Clopentixol, Patent:US3116291, CAS:982-24-1, PMID:1650428 |
| pimozide        | antidyskinesia agent, dopaminergic antagonist, H1-  | "A member of the class of benzimidazoles that is 1,3-  | pimozide, Pimozide, 1-(1-[4,4-bis(4-   | Orap, Halomonth, Opiran, pimozide, Neoperidole, pimoziidum, pimozida   | 3_STAR          | KEGG:C07566, Drug_Central:2,   |

Reset Apply Close

Figure 69 Last interactive view of the workflow showing compounds with additional information having both selected "roles" from the network view.

Extensions needed to run the workflow:

- KNIME Semantic Web/Linked Data. To read more about that extension, you might find this blog article [Integrating One More Data Source: The Semantic Web](#) interesting, too.
- RDKit KNIME Integration is used to perform substructure searches and highlighting
- Network Mining Extension is used to create a network view

## Wrapping up

We started with an OWL file containing the ChEBI ontology and went through different steps of data exploration and visualization. We showed how to read an OWL file, how to create queries in SPARQL and presented different possibilities for visualizing ontology content in an interactive composite view. With this, we learned about the Promazine compound and the biological and chemical roles of that compound. We also discovered more about similar compounds and their roles, definitions, and synonyms.

The resulting data extracted from the ChEBI ontology can be directly explored using KNIME Analytics Platform. The workflow can also be deployed to KNIME Server where an expert of a certain research domain who is maybe not a KNIME or an ontology expert can analyze the data in the WebPortal without the need to write SPARQL queries.

The workflow described in this blog post, the ChEBI Ontology Explorer can be downloaded here from the KNIME Hub.

## References

1. Whetzel PL, Noy NF, Shah NH, Alexander PR, Nyulas C, Tudorache T, Musen MA. BioPortal: enhanced functionality via new Web services from the National Center for Biomedical Ontology to access and use ontologies in software applications. *Nucleic Acids Res.* 2011 Jul;39(Web Server issue):W541-5. Epub 2011 Jun 14.
2. Hastings J, Owen G, Dekker A, Ennis M, Kale N, Muthukrishnan V, Turner S, Swainston N, Mendes P, Steinbeck C. (2016). ChEBI in 2016: Improved services and an expanding collection of metabolites. *Nucleic Acids Res.*

# Chapter 4: Lab Data

In the last chapter we want to give some examples of how KNIME can be used to deal with lab data.

## Near Infrared Spectroscopy (NIR) Data Analysis using KNIME

This story describes how to analyze near infrared spectroscopy (NIR) data with KNIME. We look at three different workflows: The first workflow is about NIR spectroscopy data pre-treatment and we can see how spectral data are preprocessed to compensate for the inherent conditions of NIR spectroscopy. The data is initially available as a Matlab file and we convert it to a csv file using GNU Octave before reading it in. The second workflow takes the preprocessed data and produces visualizations, a principal component analysis and hierarchical clustering of samples based on their corresponding spectral data. The third workflow demonstrates how to perform a similarity search for one or more spectra against a custom inhouse database based on jdx files to compare our findings e.g. to a known reference product.

## User-friendly End-to-End Lab Automation in Action

In this story we show how we integrated the SiLA standard (Standardization in Lab Automation) in KNIME Analytics Platform. We give an overview of what SiLA is, and how it integrates with KNIME. More information about where to get the Java library and where to put it is given in the component description on the KNIME Hub.

To make this usable and adaptable by others, we have encapsulated this functionality into shared components. We utilized this functionality in two use cases where we automatically retrieve and analyze data, in the first case microplate data and in the second case imaging data. The first workflow demonstrates how you can use the shared components to retrieve plate readouts (e.g., fluorescence) as XML from a SiLA server, parse and process the data with KNIME nodes, and send computation results back to the SiLA server. The second workflow shows how to compute the cell count in each image and send the results back to the SiLA server.

## What are the FAIR guiding principles and how to FAIRify your data

In this story we demonstrate a use case where KNIME was used to make data more compliant with the FAIR principles. The workflow combines laboratory data from biological assays that was previously stored in 48 individual Excel files into one large, machine-friendly data table that also integrates the information about the tested compounds. To enhance interoperability, more chemical identifiers were included and the metadata was extended by domain-specific, controlled vocabulary using programmatic access to the chEMBL and chEBI databases. In compliance with the FAIR principles, user-defined metadata was added in an interactive view. The data is now ready to be uploaded to a data repository.

# 4.1 Near Infrared Spectroscopy (NIR) Data Analysis using KNIME

By Miriam Mathea, Mireille Krier & Temesgen Dadi

Find the workflow(s) here: <https://kni.me/s/w6aGw-BcuUE95ZBp>

KNIME Analytics Platform is an open source tool that enables easy creation of workflows that can perform numerous kinds of data transformations in an intuitive way. The resulting workflows are not only easy to understand by people that have never seen the workflow, but also produce reproducible results.

In this blog post, we describe three KNIME workflows developed to work with near-infrared (NIR) spectroscopy data measured on chemical samples. NIR spectroscopy measures absorption (or transmittance) of light with wavelengths between 780 and 2500nm.

This method, which is non-intrusive and requires comparatively little sample preparation, is applied in a wide range of fields such as agriculture, product monitoring, polymer engineering, biomedicine, pharmaceutical industry, and environmental science. The method's low sensitivity to minor experimental constituents makes it necessary to set up calibrations that require many samples. Additionally, raw data need to be preprocessed to account for the physical property of the sample particles such as particle size, density and scatter, before proceeding to quantitative analysis.

In this article we look at three workflows for the pre-treatment and analysis of NIR spectroscopy data.

- The first workflow, NIR Spectroscopy data pre-treatment, deals with data treatment and we look at how spectral data are preprocessed to compensate for the inherent conditions of NIR spectroscopy,
- The second workflow, Visualization, Clustering, and PCA analysis of Preprocessed Spectral Data, produces visualizations, PCA, and hierarchical clustering of samples based on their corresponding spectral data,
- The third workflow, Similarity Search Using Inhouse Database, demonstrates how to perform a similarity search for one or more spectra against a custom inhouse database.

## 1. NIR Spectroscopy data pre-treatment

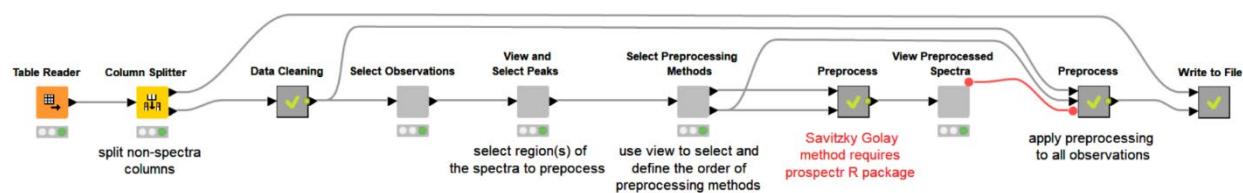


Figure 70 Workflow I - NIR spectroscopy data pre-treatment.

Let's start by reading our NIR dataset, which is a challenge NIR spectra dataset obtained from [Github](#). The dataset is available as a matlab file and we used GNU Octave to convert it to a CSV format before reading it in KNIME Analytics Platform. The samples came from straws and are associated with the sugar potential of each straw. A straw's value for energy production is highly tied to its sugar potential and is determined by a complex process with the help of an enzymatic kit. We took a subset of the dataset for the purpose of our blog post. In particular, we took 357 samples measured by NIR in the range 400-2498nm.

| Row ID | SampleName   | Glucose | spec.400 | spec.402 | spec.404 | spec.406 |   |
|--------|--------------|---------|----------|----------|----------|----------|---|
| Row0   | T?07-127-A-5 | 0.171   | 0.568    | 0.589    | 0.611    | 0.628    | 0 |
| Row1   | T?07-128-A-5 | 0.189   | 0.543    | 0.566    | 0.588    | 0.607    | 0 |
| Row2   | T?07-129-A-5 | 0.173   | 0.532    | 0.556    | 0.578    | 0.595    | 0 |
| Row3   | T?07-131-A-5 | 0.18    | 0.612    | 0.631    | 0.65     | 0.666    | 0 |
| Row4   | T?07-132-A-5 | 0.199   | 0.577    | 0.598    | 0.619    | 0.638    | 0 |
| Row5   | T?07-133-A-5 | 0.286   | 0.553    | 0.572    | 0.591    | 0.609    | 0 |
| Row6   | T?07-134-A-5 | 0.167   | 0.544    | 0.568    | 0.593    | 0.613    | 0 |
| Row7   | T?07-135-A-5 | 0.179   | 0.547    | 0.57     | 0.593    | 0.61     | 0 |
| Row8   | T?07-129-A-5 | 0.160   | 0.574    | 0.594    | 0.611    | 0.621    | 0 |

Figure 71 Raw Spectral Data containing the sample name, the sugar potential, and the NIR spectra in the range 400-2498nm.

The first thing we want to do is visualize the spectra. Since it is difficult to plot and visualize the spectra of all 357 samples, we will select a few interactively. Doing this with a component with a column filter configuration is a good fit.

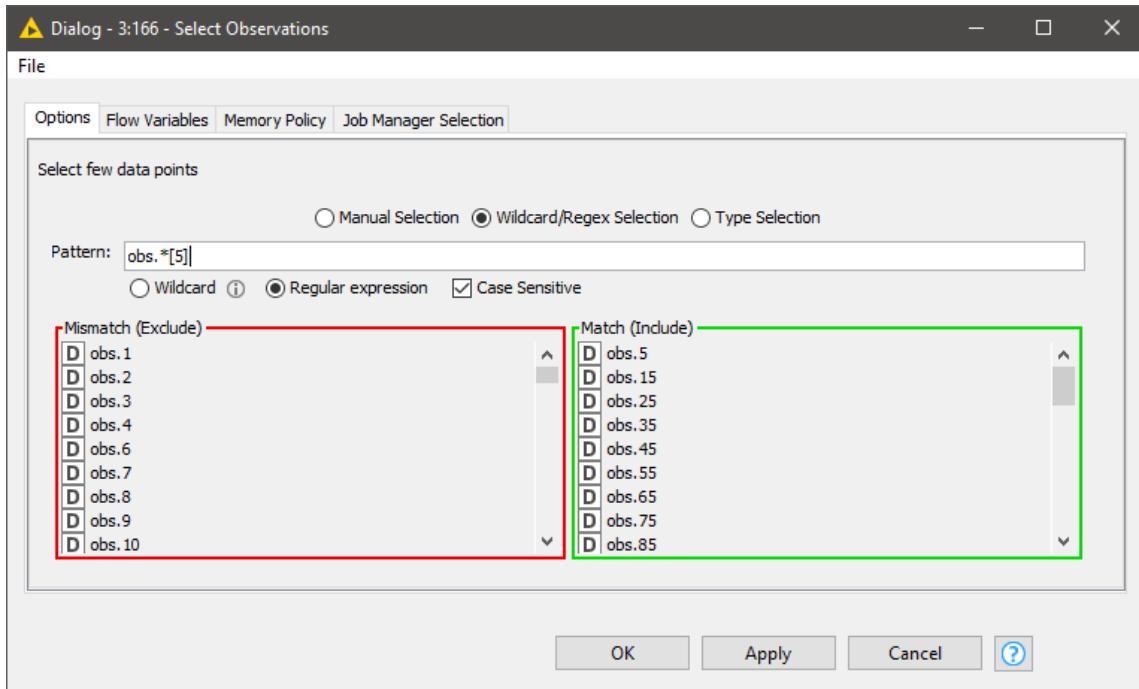


Figure 72 Component configuration dialog for selecting observations

Once we have selected representative observations we can visualize their spectra and box-select the interesting regions that will be used for preprocessing. In this example we chose to use the region from 1500-2000nm for our preprocessing. The selection is highlighted as a thick line.

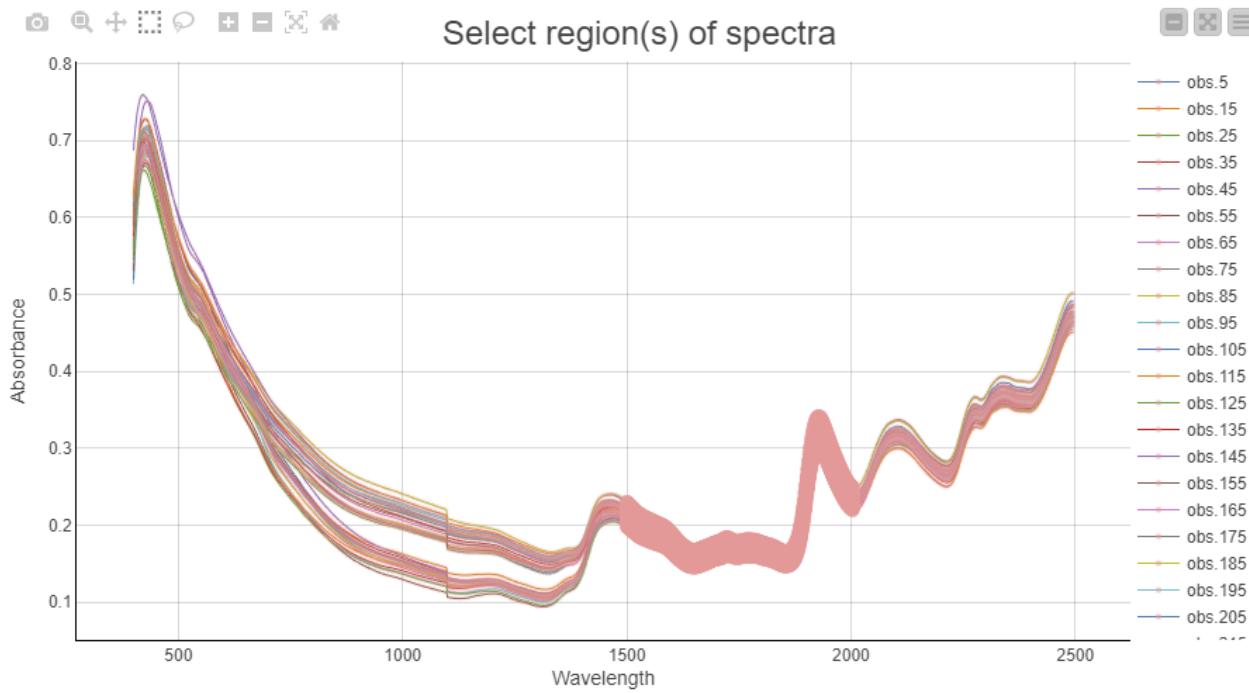


Figure 73 Component view for selecting region(s) of spectra. One can box-select interesting regions for further analysis.

The next step is to decide which preprocessing methods to use and in what order they should be applied to the data.

In general, the most widely used preprocessing techniques can be classified into two categories. In the first category are the scatter-correction methods and in the second, the spectral derivatives. As representatives from the first category we selected Standard Normal Variate (SNV) and Normalization (Centering and Scaling) and for the second category Savitzky-Golay (SG) polynomial derivative filters<sup>1,2,3</sup>.

Scatter correction methods are used to remove undesired spectral variations due to light scatter effects and variations in effective path length. SNV is a very simple method for normalizing spectra to correct for light scatter<sup>1</sup>. It is calculated by subtracting the mean of spectrum from individual values and dividing the result by standard deviation of the spectrum. Normalization transforms the spectra matrix to a matrix with columns with zero mean (centering), unit variance (scaling) or both (auto-scaling). In contrast to SNV, which operates row-wise, normalization operates column-wise. The Savitzky Golay method fits a local polynomial regression on a signal using an equidistant width<sup>2</sup>.

Select preprocessing methods

|                                     | Method Name | Method Desc.                  | Required Package      | ApplicationOrder |
|-------------------------------------|-------------|-------------------------------|-----------------------|------------------|
| <input checked="" type="checkbox"/> | Centering   | Centering                     | None                  | 1                |
| <input checked="" type="checkbox"/> | Scaling     | Scaling                       | None                  | 2                |
| <input checked="" type="checkbox"/> | SNV         | Standard Normal Variate (SNV) | None                  | 3                |
| <input checked="" type="checkbox"/> | Savitzky    | Savitzky-Golay filtering      | prospectr (R package) | 4                |

Showing 1 to 4 of 4 entries

#### For Savitzky Golay:

|                             |                   |                            |
|-----------------------------|-------------------|----------------------------|
| Polynomial order (1-4)<br>3 | Window Size<br>11 | m-th Derivative (0-3)<br>2 |
|-----------------------------|-------------------|----------------------------|

Figure 74 Component view for selecting preprocessing methods and defining their order of application.

The selection of preprocessing techniques is done using the interactive view of a dedicated component. Methods can be selected and their order can be defined by editing the table. If the Savitzky Golay method is selected, its three additional parameters (polynomial order, window size and m-th derivative) can be adjusted.

Now, if we execute and open the view of the last component, we can clearly see the effects of our selected preprocessing methods on the data.

Once we verified that the preprocessing methods are working as expected, the same transformation will be applied to all samples to be preprocessed. Then we export the preprocessed data to disk for the next steps of spectral analysis

## Preprocessed Spectra

Centering  $\Rightarrow$  Scaling  $\Rightarrow$  SNV  $\Rightarrow$  Savitzky ( $p=3$ ,  $w=11$ ,  $m=2$ )

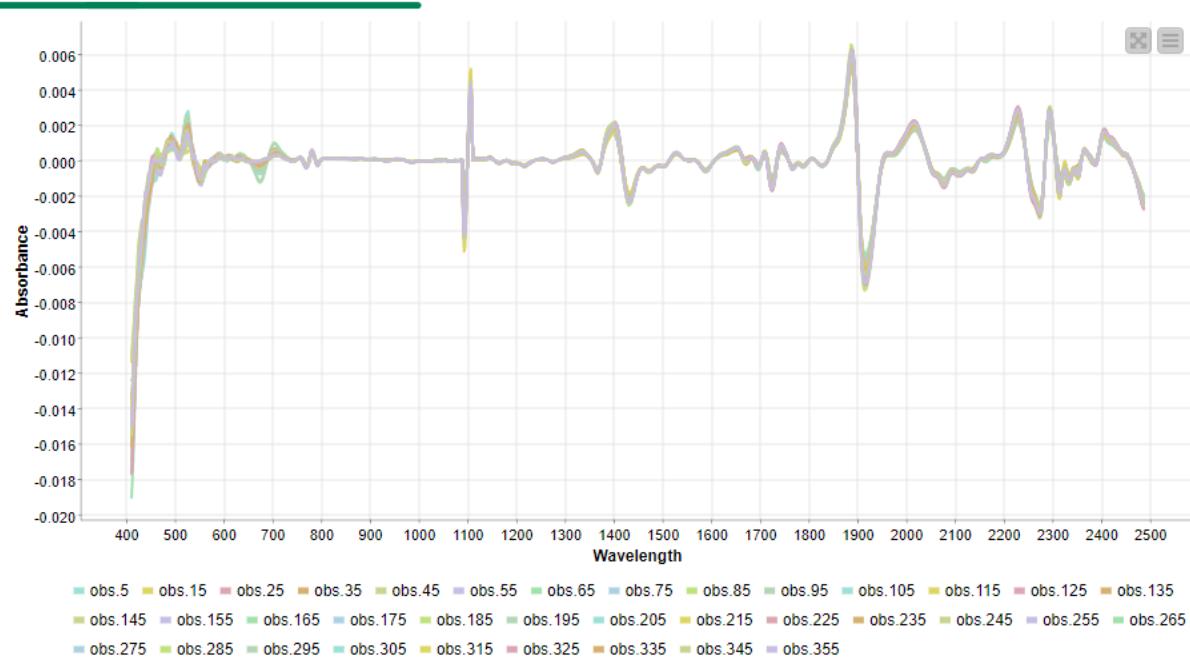


Figure 75 Spectra after preprocessing. After applying centering, scaling, SNV, and Savitizky Golay filtering the undesired spectral variations are removed.

## 2. Visualization, Clustering, and PCA analysis of Preprocessed Spectral Data

Finding patterns in the data and deriving meaningful insights from this is among the strong features of the KNIME Analytics Platform. In our case this can be done through visualization, principal component analysis and hierarchical clustering. The second KNIME workflow makes use of the preprocessed data from above and does exactly these three things. We are still using the same dataset containing samples of straws associated with their sugar potential. The dataset was created to determine if NIR spectroscopy can be used to determine the sugar potential of straws instead of the complex process that involves enzymatic degradation.

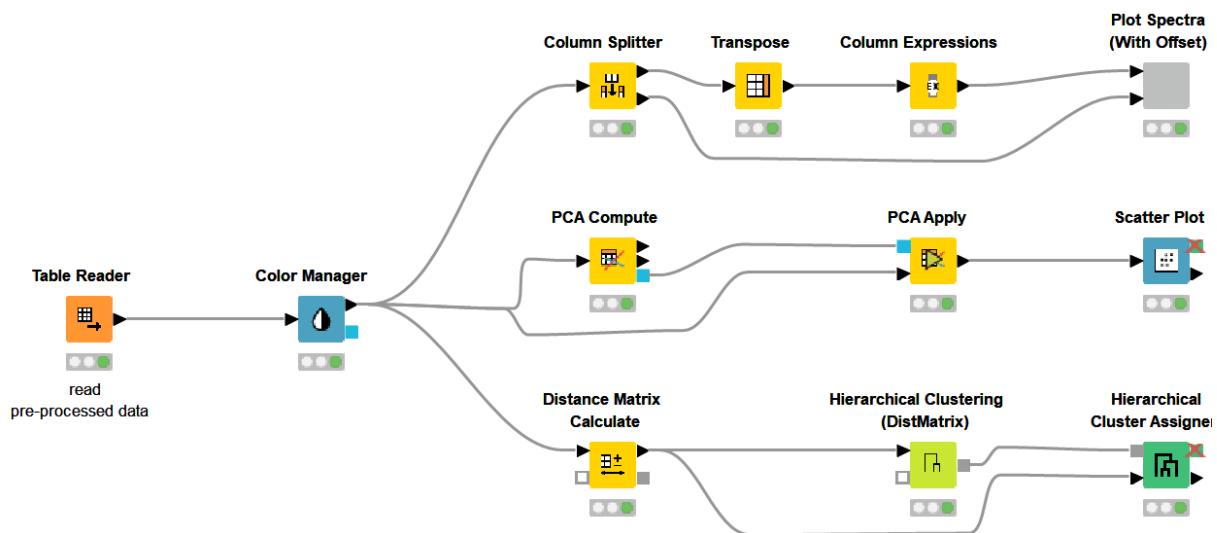


Figure 76 Workflow II - Visualization, Clustering and PCA analysis of Preprocessed Spectral Data.

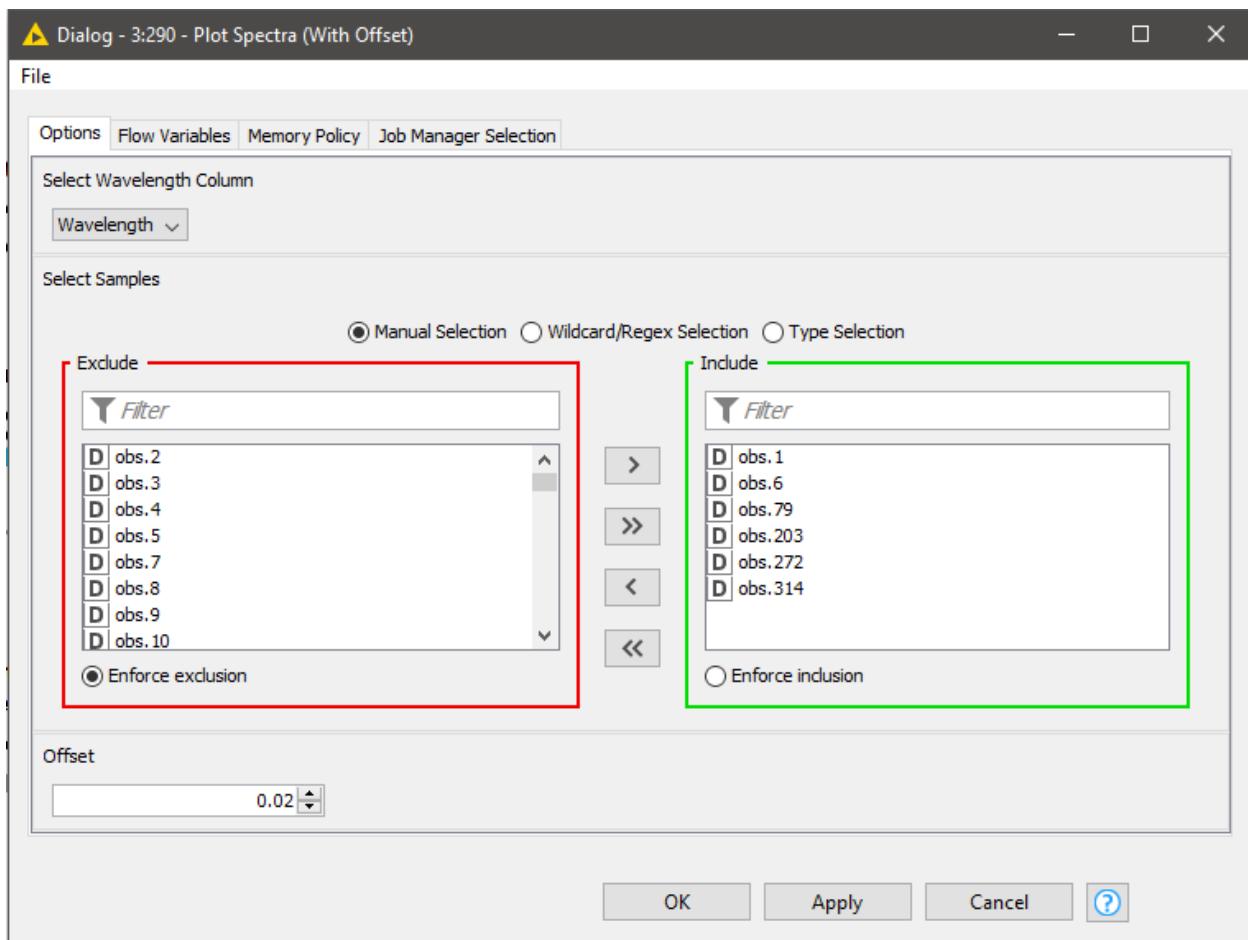


Figure 77 Configuration dialog of Plot Spectra (with offset) component.

In spectroscopy, plotting multiple spectra as line plots for visual inspection of differences is a common task. To make the comparison easy, different spectrums are plotted with a fixed offset with the goal of avoiding overlapping lines. Using a configurable component (see component configuration dialog - Figure 77), where we can select the spectra to show in the plot and set the offset, plotting spectra can be done easily. The resulting plot is shown in Figure 78.

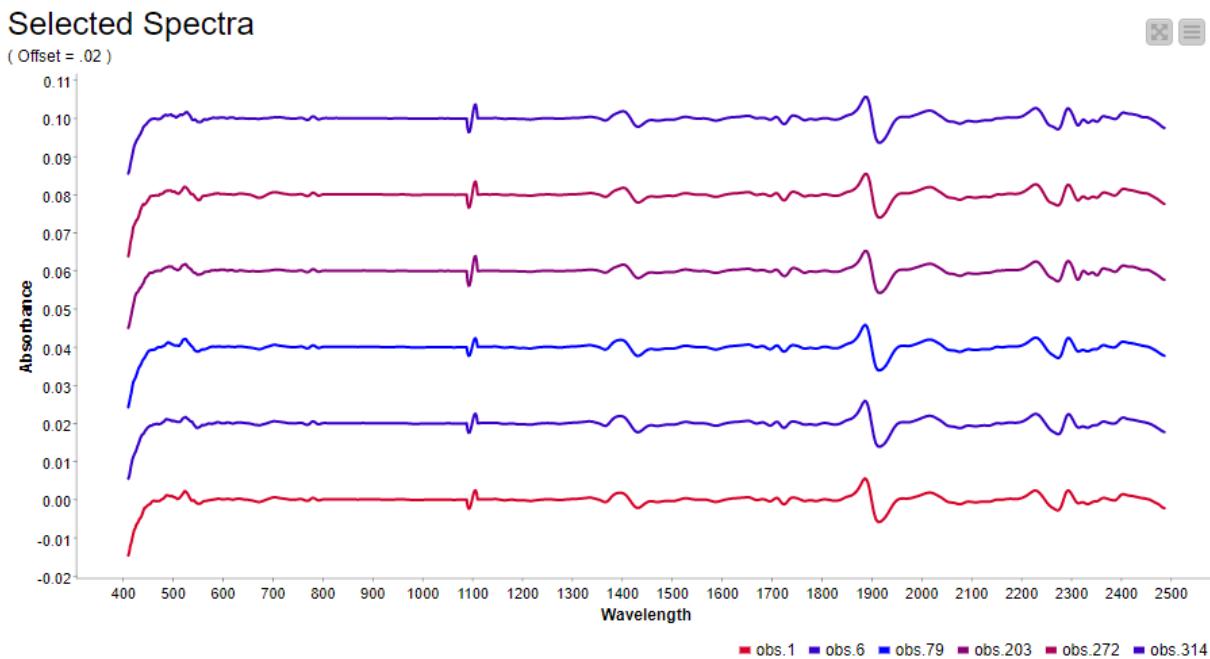


Figure 78 Selected spectra plotted using the Plot Spectra (with offset) component (offset = 0.2).

Hierarchical clustering in KNIME is simplified via the Distance Matrix Calculate node where we can compute the distance of choice between samples. Using a combination of just three nodes we were able to observe the clustering that is associated with the level of glucose in the straw samples. Samples are colored according to their glucose content measured by an enzymatic degradation and we can see that samples with similar glucose content tend to cluster together. This suggests NIR Spectroscopy has the potential to discriminate between good and bad straw samples (Figure 79).

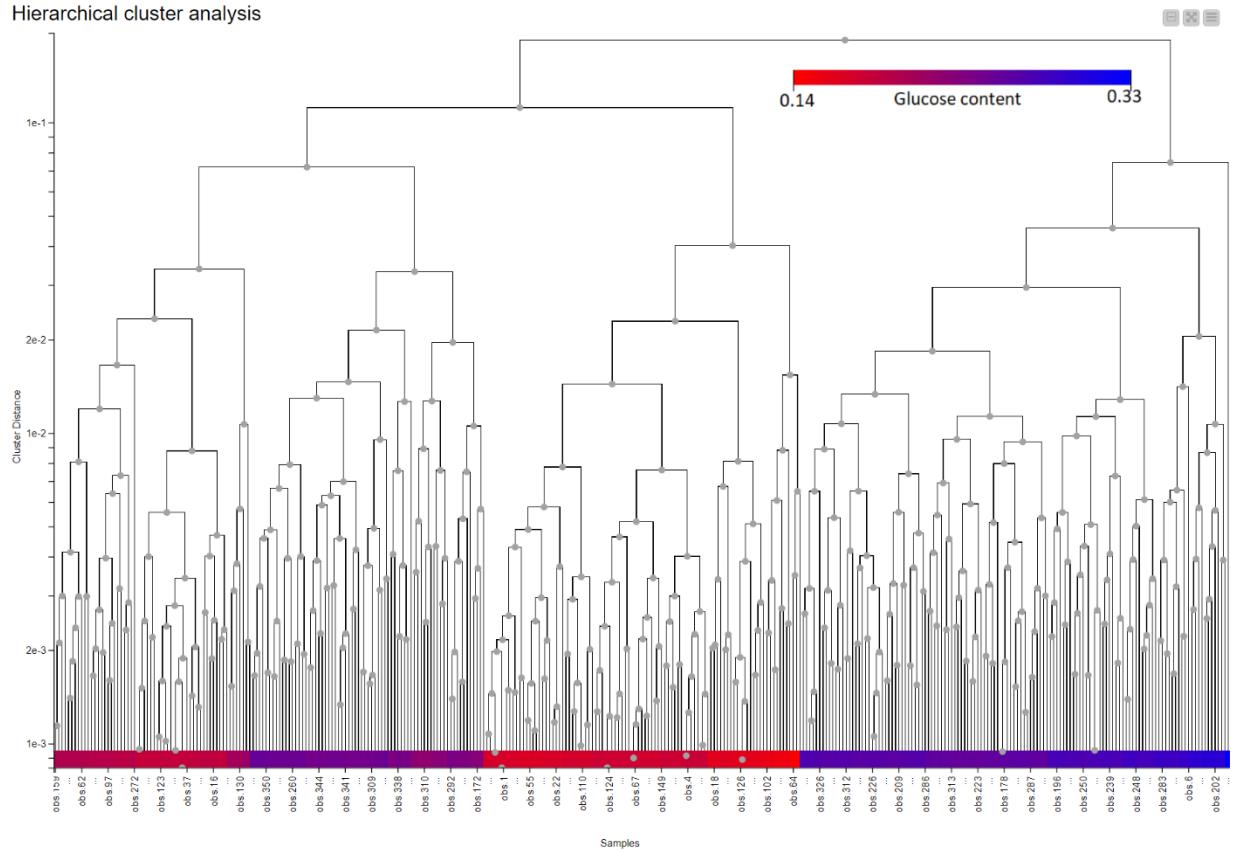


Figure 79 Hierarchical clustering of straw samples (colors represent sugar potential of straws).

Similar to the Hierarchical clustering, PCA Analysis has the potential to reveal patterns and detect outliers. It can be performed easily using a couple of nodes in KNIME Analytics Platform. Figure 80 shows the results of a simple PCA analysis on the straw samples using the preprocessed spectral data. There is a clear separation between the straws with high potential for sugar (blue) and those with low levels of sugar (red). The PCA plot is the result of using the whole NIR range as input. We have a separation between samples, in terms of sugar level, across PC2 instead of PC1. This led us to think there is a more pronounced second pattern in the data which is unrelated to sugar level. When we went back to our preprocessing workflow and looked at where the samples show differences, we found two regions (700nm-1400nm and 1500nm-2000nm) with notable differences between the samples. After trying both regions, we picked wavelength region 1500nm - 2000nm and obtained the result on the right where we have a better clustering along the main principal component (PC1).

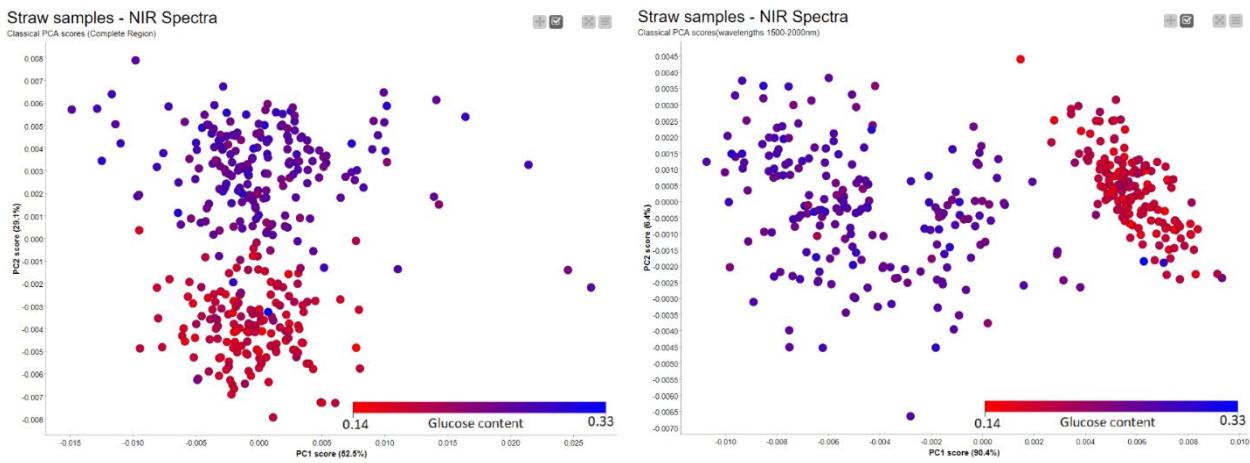


Figure 80 PCA analysis of straw samples (colors represent sugar potential of straws) using the whole NIR spectra range (left) and using NIR region 1500nm - 2000nm (right).

### 3. Similarity Search Using Inhouse Database

One of the applications of NIR spectroscopy is confirming the authenticity and quality of a chemical or food product by comparing it to a known reference product. At the heart of such applications is a method to measure the similarity between two spectra. Common methods in NIR spectroscopy are the correlation coefficient, the Euclidean, or the Mahalanobis distance. With this simple workflow, we will showcase how we can perform a database search of a spectrum using these (commonly used) distance measures.

#### III. Similarity search (data source: OSDB database - <http://osdb.info/>)

**NOTE:** The data here (JDX files obtained from OSDB database) is **IR** spectra data.  
But the procedure works for any kind of spectra including **NIR** spectra

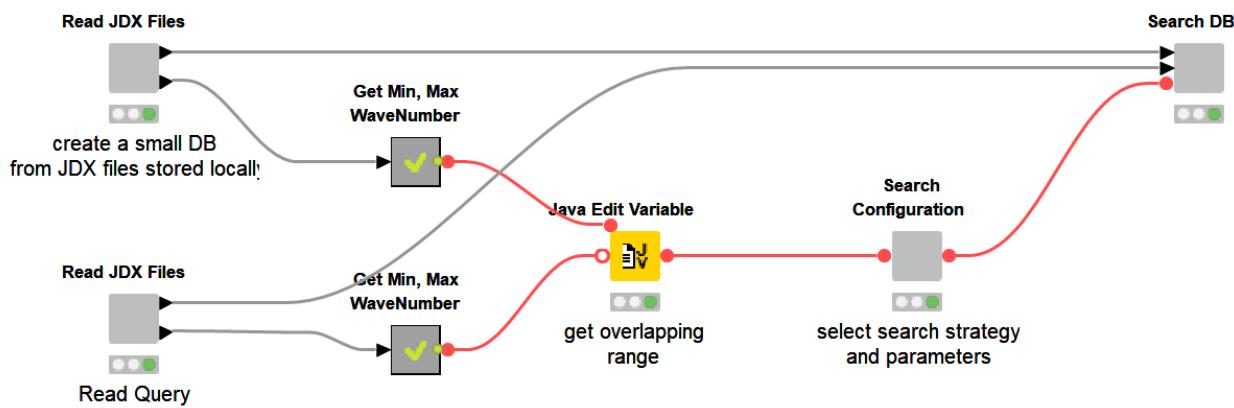


Figure 81 Workflow III - Similarity Search of Spectra using Inhouse Database.

Storing spectral data in JCAMP-DX (.jdx) files is a common practice and many spectrometers directly produce data as .jdx files. We created a component that builds a spectral database by reading .jdx files contained in a folder, where each file represents a compound. After parsing

individual .jdx files, the global wavelength ranges are detected and missing values (readings on a given wavelength that are present in compound A but not in compound B) are filled via linear interpolation. The result is a table where compounds are arranged row-wise and wavelengths in columns.

In this example, we are using 20 JDX files downloaded from the [Open Spectral Database](#) representing 20 different compounds to create our spectral database. The workflow can be adjusted to use a different list of JDX files with the goal of creating an own database. Figure 82 shows the list of compounds (files) that make up the database and the database itself shown as a KNIME table. This is a simple way of creating a database. For real world cases robust databases should be designed. For example, instead of a single spectral entry per compound, we can have multiple spectra representing multiple sources of variability associated with the product and the manufacturing process.

The screenshot shows a KNIME node titled "Port 1 - 0:234 - Create DB from JDX Files (create a small DB)". On the left, there is a tree view under "DB\_JDX\_Files" containing 14 JDX files: Adamantane.jdx, Aspirin.jdx, Atenolol.jdx, Benzaldehyde.jdx, Bromobenzene.jdx, Butanone.jdx, Caffeine.jdx, Chlorobenzene.jdx, Ethanal.jdx, Ethanamide.jdx, and others. To the right is a table titled "default" with 20 rows and 1702 columns. The table has a header row with "Row ID" and several wavelength columns labeled 600, 602, 604, 606, and 608. Below is a sample of the data:

| Row ID         | D 600  | D 602  | D 604  | D 606  | D 608  |
|----------------|--------|--------|--------|--------|--------|
| DB_Adaman...   | 97.745 | 97.745 | 97.745 | 97.745 | 97.745 |
| DB_Aspirin     | 83.35  | 83.35  | 83.35  | 83.35  | 83.35  |
| DB_Atenolol    | 73.4   | 73.4   | 73.4   | 73.4   | 73.4   |
| DB_Benzalde... | 52.5   | 53.5   | 54.6   | 55.6   | 56.5   |
| DB_Bromoben... | 52.5   | 53.5   | 54.6   | 55.6   | 56.5   |
| DB_Butanone    | 42.1   | 43.15  | 44.65  | 46.1   | 46.75  |
| DB_Caffeine    | 94.25  | 94.25  | 94.25  | 94.25  | 94.25  |
| DB_Chlorobe... | 51.4   | 51.75  | 53.35  | 54.6   | 55.2   |
| DB_Ethanal     | 46.9   | 48     | 48.85  | 49.6   | 51.4   |
| DB_Ethanamide  | 39.6   | 39.6   | 39.6   | 39.6   | 39.6   |

Figure 82 The list of compounds in the database and the content of the database.

The next step is to read the query files that need to be searched against the database. We used a similar procedure as reading the database entries to read the queries as well. We have the spectra of Ibuprofen and three different readings of paracetamol (Figure 83).

The screenshot shows a KNIME node titled "Port 1 - 0:233 - Read Query (JDX Files)". On the left, there is a tree view under "Table" containing 4 rows. To the right is a table titled "default" with 4 rows and 1751 columns. The table has a header row with "Row ID" and several wavelength columns labeled 500, 502, 504, 506, 508, 510, 512, and 514. Below is a sample of the data:

| Row ID             | D 500  | D 502  | D 504  | D 506  | D 508  | D 510  | D 512  | D 514 |
|--------------------|--------|--------|--------|--------|--------|--------|--------|-------|
| Query_Ibuprofen    | 99.923 | 99.415 | 98.907 | 98.399 | 97.891 | 97.383 | 96.876 | 86.9  |
| Query_paracetamol1 | 59.072 | 59.315 | 65.595 | 79.559 | 89.75  | 81.447 | 65.607 | 64.4  |
| Query_paracetamol2 | 75.359 | 75.359 | 75.359 | 75.359 | 75.359 | 75.359 | 75.359 | 75.3  |
| Query_paracetamol3 | 0.001  | 0.001  | 0.001  | 0.001  | 0.001  | 0.001  | 0.001  | 0.001 |

Figure 83 Spectra of compounds to be searched against a database.

Once both the database and query tables are ready, we can define our search strategy via the configuration dialog of the Search Configuration component. This includes:

1. The spectral region where the comparison should focus
2. How many hits we should display per query
3. Which similarity metrics to use

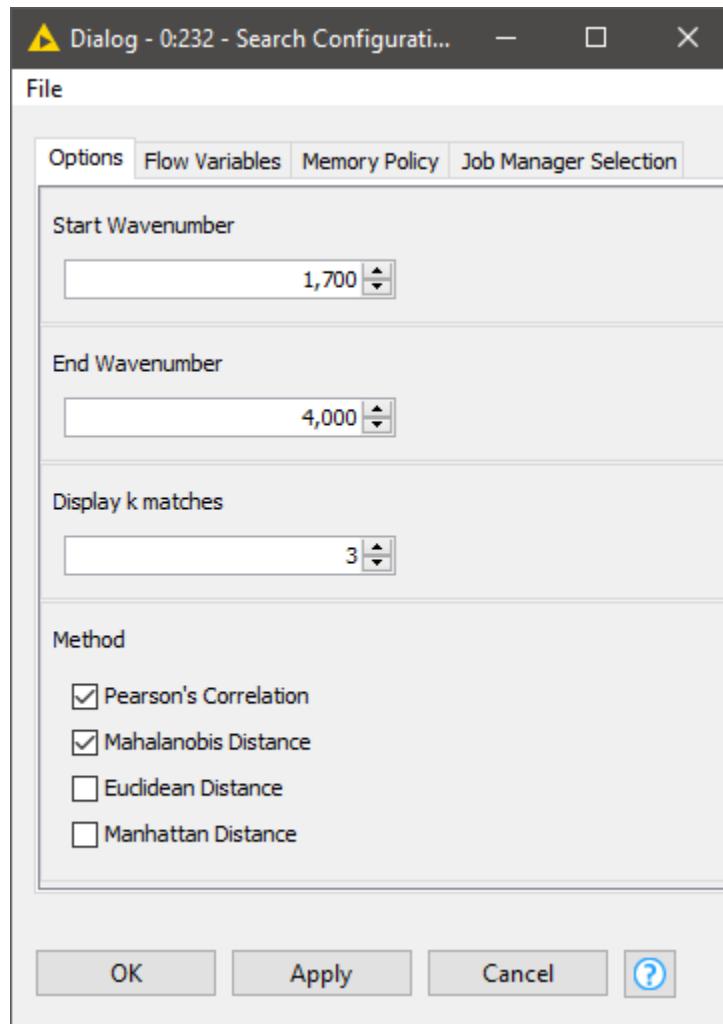


Figure 84 Search Configuration. The user can select the spectra region, the hits displayed per query, and the similarity matrices to use.

Executing the Search DB component and opening its view results in a list of top-k matches per query, or multiple lists if we selected more than one more than one similarity metrics. The list contains k matches irrespective of how good the matches are. We can filter the matches using the interactive range slider to enforce a certain degree of similarity index according to the search method in question.

The example result shown in the figure below is a result of searching two different paracetamol spectra against our tiny database using Pearson's correlation and Mahalanobis distance as similarity metrics. Not surprisingly, the database entry Paracetamol appeared as the best hit for both spectral queries and all search strategies.

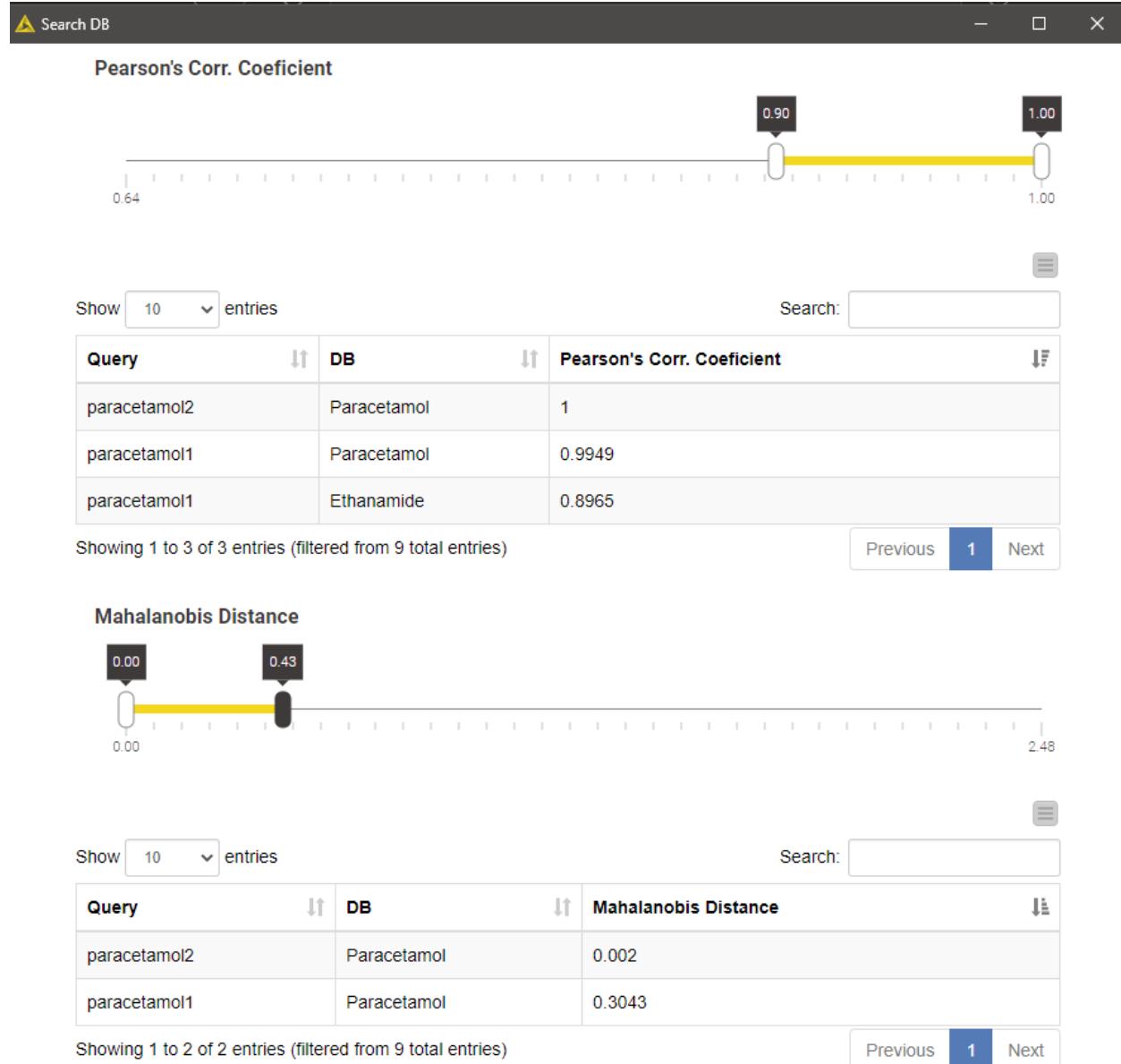


Figure 85 Search results obtained using Pearson's correlation and Mahalanobis Distance methods. Results can be sorted in the table and can be filtered using the provided range slider. Adjusting for higher correlation or lower distance values leaves the correct hits of paracetamol.

## Conclusion

We have shown three different KNIME workflows for NIR spectral data analysis. We have created a workflow that treats spectral data to account for experimental artifacts and method-related biases. Then we performed a PCA analysis and hierarchical clustering on the resulting data to determine if NIR methods can be used as an alternative way of measuring the sugar potential of straws. The results look promising as both our PCA analysis and hierarchical clustering results group straws with similar levels of sugar together. The PCA analysis also revealed the wavelength region 1500nm - 2000nm has a particular importance in relation to the sugar level of the straw samples. This is helpful information for further analysis such as creating a regression model that can actually predict the sugar potential of a straw from NIR spectral data. Finally we showed how an inhouse database can be created directly from jdx files and searched using different similarity search strategies with the aim of identifying a product or checking the authenticity of a product.

Taken together, we've demonstrated how KNIME Analytics Platform can be a powerful tool for typical spectra analyses. The workflows we provided can be downloaded from the KNIME hub and be adjusted to your own data.

## 4.2 User-friendly End-to-End Lab Automation in Action

By Jeany Prinz & Stefan Helfrich

Find the workflow(s) here: <https://kni.me/s/w6aGw-BcuUE95ZBp>

Many life science companies have important digitalization initiatives that incorporate end-to-end integration. The goal of such initiatives is to improve experimental reproducibility, reduce errors, increase productivity, and enhance the generated data through contextualization. However, the effort required to implement these is high, due to contending with inconsistent infrastructures as well as a diverse set of integration interfaces. This hinders an effective implementation; it often seems easier to stick to doing everything manually. A solution to this problem is to integrate standards and to simplify end-to-end integration.

### Standardized lab automation with KNIME Analytics Platform

We show how we integrated the SiLA standard (Standardization in Lab Automation) in KNIME Analytics Platform through a collaborative effort of [siobra](#), [Biosero](#), [HDC](#), KNIME, and the [SiLA](#) consortium. SiLA enables the automation and digitizing of scientific laboratories through free and open data standards and systems communication.

We utilize the SiLA integration in two use cases that automatically analyze

- microplate data and
- imaging data

These workflows are useful beyond these two use cases. The SiLA standard can be used in a lot of different scenarios, and the workflows easily adjusted and applied to diverse projects. With the demonstrated integration we enable lab scientists to perform their own analyses by building reproducible workflows to automatically retrieve their data, blend them with other data sources, and carry out different kinds of post-processing.

You can find [videos](#) of the workflows we built for the two use cases in action on Youtube. The video demonstrates how KNIME workflows can automatically retrieve and analyze laboratory data in one simple and intuitive environment for end-to-end data science.

### SiLA for open connectivity in lab automation

The SiLA Consortium is a non-profit membership organization that brings together lab users, vendors, and start-ups to define standards that permit information interchange in the lab. SiLA's mission is to establish international standards that create open connectivity in lab automation. It provides a framework for free and open system communication and data standards to connect scientists with the data that matters, and it is supported by some of the biggest and most innovative names in the industry. It is built using a message-driven architecture based on HTTP/2,

thereby supporting the delivery and management of data from a wide range of laboratory devices. Devices expose their properties and commands (i.e., operations) to the network via the so-called SiLA 2 server. A SiLA 2 client can then establish a network connection to the server and initiate command execution on the device (Por M. et al. 2020).

## How does SiLA integrate with KNIME?

The SiLA consortium and its members maintain reference implementations of the SiLA 2 standard for various programming languages. Among others, an implementation for Java is available [here](#) under MIT license. Since the sources are publicly available, siobra was able to repackage this reference implementation and implement a light-weight wrapper that can readily be used from KNIME Analytics Platform. Early in the project we decided to go for a proof-of-concept implementation that uses Java Snippet nodes utilizing the aforementioned wrapper. To make this usable (but also adaptable) by others, we have encapsulated this functionality into shared components. Please note that in order to minimize the size of the components, the JAR file containing the wrapper is available separately. This has also allowed us to iterate rapidly on the integration together with the other collaborators, e.g. by replacing one instance of the wrapper that is used in all components. It furthermore allows tech-savvy users to use components as blueprints for connecting to their lab devices via SiLA (without the need to recompile and package sources).

We have created three shared components to handle different tasks: one to extract binary data such as images from the SiLA Server, one to retrieve XML data, and one to send the results back to the SiLA Server. The three components can be found on the KNIME Hub for easy reuse.

- [Get Binary Stream from SiLA Server](#)
- [Get XML from SiLA Server](#)
- [Send Results to SiLA Server](#)



Figure 86 Shared components to retrieve binary (e.g. images) and XML data from SiLA Server and send the results back to the server.



## Use Case: Automatic Analysis of Microplate Data

The first example workflow demonstrates how you can use the new shared components to retrieve plate readouts (e.g., fluorescence) as XML from a SiLA server, process the data with KNIME nodes, and send computation results back to the SiLA server. You can download the workflow “Get Data from SiLA 2 Server and Send Back Results” from the KNIME Hub here.

The workflow comprises four simple steps as can be seen in Figure 87.

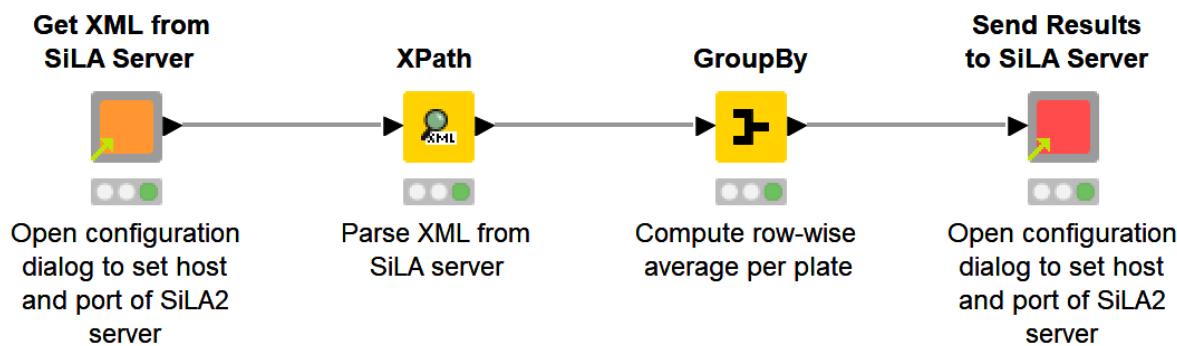


Figure 87 Workflow to prompt measuring data on a microplate, extracting and processing the data and sending the results back to the SiLA Server.

### 1. Get XML from SiLA Server

We use the shared component to retrieve microplate readouts. A connection to the SiLA 2 Server is established based on the given host name and port and performing the measurement is triggered. The data is then sent back to KNIME as XML.

### 2. Parse XML

The retrieved data is parsed into a table using the XPath node, which allows to perform XPath queries to create a data table in an intuitive way.

### 3. Process the data

In this simple example, we perform a row-wise average per plate using the GroupBy node to process the data. More complex operations including interactive visualizations would be possible in this step as well.

### 4. Send Results to SiLA Server

In the last step, we send the data back to the SiLA 2 Server using the provided shared component. In addition to the host name and port we also have to configure the column with keys and values.

## Use Case: Automatic Analysis of Imaging Data

The second example workflow demonstrates how to analyze data from an imaging experiment that comprises multiple, potentially multi-channel or time-resolved, images. That is, instead of

well-structured XML from a plate reader, we receive binary data in the form of images from a SiLA Server. In this example, we do not get each image as an individual file, but rather collected in a compressed archive (e.g., ZIP). Once the contents have been extracted, we use the KNIME Image Processing Extension to derive information from the images.

In Figure 88 below, you can see the entire workflow, comprised of the following steps:

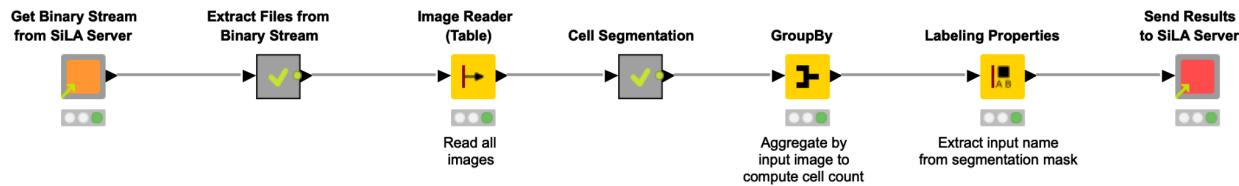


Figure 88 Workflow to receiving images, extracting cell counts, and sending the results back to the SiLA Server.

The steps of the workflow are as follows:

#### 1. Get Binary Stream from SiLA Server

Request and receive a binary stream from SiLA Server. This automatically creates a Binary Object column that can be handled with generic KNIME functionality (see below).

#### 2. Write Binary Stream to Archive File and Extract

Take the Binary Object column and write the content into a file in a dedicated temporary directory. Since we expect the binary stream to contain an entire ZIP archive, we create and extract such an archive. A table with file paths (in this case: images) is returned from the metanode.

#### 3. Read Images

Use KNIME Image Processing to read all received images into a column of a dedicated Img type.

#### 4. Cell Segmentation

This metanode implements a simple, intensity-based cell segmentation approach. That is, the cells in each input image are identified based on the fact that they show an increased intensity in comparison to the background of the images.

#### 5. Computing Cell Count per Image

Here, we aggregate information from all input images in order to compute the number of cells per image. This information can, for instance, be used to determine if the cells have reached the desired density to continue with the experiment.

#### 6. Send Results to SiLA Server

Using the aforementioned component, we send back the cell counts per image.

## **Summary**

### Reproducible Workflows for Automatic Retrieval and Analysis of Lab Data

We learned today how to create reproducible workflows that automatically retrieve and analyze laboratory data. All this can be done in one simple and intuitive environment for end-to-end data science.

The collaboration of siobra, Biosero, HDC, KNIME, and the SiLA consortium worked together to integrate the SiLA standard in KNIME Analytics Platform. In this article, we showcased the new SiLA integration in two use cases: one automatically analyzes microplate data and the other automatically analyzes imaging data. You can now simply download the shared components and workflows and apply them to your own data. Try them out yourself!

## **References**

Porr, Marc, et al. "Implementing a digital infrastructure for the lab using a central laboratory server and the SiLA 2 communication standard." *Engineering in Life Sciences* (2020).

## 4.3 What are the FAIR guiding principles and how to FAIRify your data

By Alice Krebs

Find the workflow(s) here: <https://kni.me/w/Py9va0SQesbauPwS>

When research data cannot be found, when access, interoperability, and reuse are impaired, the impact can be significant, hampering data-driven innovation and knowledge discovery, jeopardizing the success of collaborations, and binding budget and resources in areas not contributing to the competitive edge. The FAIR guiding principles, which were published in 2016, aim at facilitating maximum data reuse and are now gaining more and more attention in academic research as well as in industry, with many gatekeepers such as funding organizations and publishers now enforcing their implementation.

In this article, we show how KNIME Analytics Platform can contribute to creating FAIR data sets, from reading various data formats, to restructuring data, and extending the metadata to meet the FAIR metadata standards with controlled vocabulary and user-defined information.

### What are FAIR data?

Maybe you have already stumbled across the term FAIR data in various contexts? The concept of FAIR has been around since 2016 (Wilkinson et al. 2016) and has gained substantial traction not only in the academic research environment (especially in health and biomedical research), but also in industry.

FAIR is an acronym and stands for Findable, Accessible, Interoperable, and Reusable. Behind those four words is a set of principles that focuses on ensuring that research data and objects are available and reusable and thereby will be actually reused. Some estimates say that up to 80% of results from publicly funded research is never used again, e.g. included in a meta or re-analysis. Considering the technical possibilities we have these days to deal with data, this is a massive waste of resources. The reason for impaired reuse lies often in insufficient documentation of data and metadata, or formats that are not machine friendly. Leaving these resources untouched negatively impacts collaborative research activities, costs, and resources. Hence, ensuring that data is reusable also ensures the value of these resources is maximized as much as possible.

The ultimate aim of following FAIR principles is that machines as well as humans can find, access, interoperate, and reuse each other's data stemming from research or other objects. That extra bit of value lies for sure in their emphasis on machine-actionability, which has been lacking in previous initiatives, for example like open source. Gatekeepers like publishers or funding organizations have picked up the FAIR principles to various extents and mandate their application.

## **How you can use KNIME to help make your data FAIR**

This is the more interesting issue here. Well-structured, machine-readable, and documented data are not only important for people working with big data or databases: This also applies to people that maybe don't feel addressed in the first place because they only handle a manageable amount of data. These are often present in unstructured Excel spreadsheets, maybe with some manual analysis right next to the 'raw' data. KNIME Analytics Platform is a great tool for re-structuring these kinds of data and bringing them into a machine-friendly, interoperable form. KNIME Analytics Platform can read numerous different file formats, extract data, and transform them. It can concatenate tables or combine/join them on user-defined criteria, and all of that without - or only a minimal need for - coding. It can be used to extend the metadata with controlled vocabulary and even automatically write the output to respective repositories.

The workflow, FAIR data with KNIME, presented here represents a use case from academic research, more precisely from toxicological testing. It uses data derived from a cell-based *in vitro* assay conducted on 96-well plates, in which the effects of substances on the cells are evaluated using automated microscopy and image analysis. This workflow is in principle applicable to any assay that is conducted with a fixed plate layout.

### Concatenating the input data

To avoid confusion, we want to quickly specify the terms data and metadata for the ones who are not familiar with them: Data refers to a piece of information, e.g. numbers from measurements or quantification, images or observations. Metadata refers to data describing the data. Metadata specifies the relevant information about the data which helps in identifying the nature and feature of the data.

The data that have been used in this example are data from the image-based NeuriTox assay (also called UKN4), which assesses neurite outgrowth of human dopaminergic neuronal cells in response to substance treatment. In this assay, a proof of concept substance library assembled by the U.S. National Toxicology Program (NTP) was screened, and the library comprised 80 (potentially) neurotoxic substances. If you are interested in the outcome, the results were published in 2018 in this paper. All the data files used here have been kindly provided by the Leist Lab at the University of Konstanz, Germany.

The actual raw data are images and won't be shared here, but we're going to keep it as raw as possible. At this stage that's the numeric outcome of automatic image analysis, which has been stored in 48 individual Excel files of a defined format (see Figure 89). The technical replicates are represented by the three blocks of columns, while the three different endpoints are in row-wise blocks (and to answer the attentive reader's question: Yes, the technical replicates are on separate plates. Yes, that's unusual, but it has been really thoroughly tested and validated and it does not affect the variability of the outcome).

| 2010-11_Full_Team_10_11_10_11 |      |      |                            |     |     |                   |      |      |                     |      |      | 2010-11_Full_Team_10_11_10_11 |     |     |                     |     |     |                   |      |      |                     |      |      | 2010-11_Full_Team_10_11_10_11 |     |      |                     |     |      |                   |     |      |     |      |     |
|-------------------------------|------|------|----------------------------|-----|-----|-------------------|------|------|---------------------|------|------|-------------------------------|-----|-----|---------------------|-----|-----|-------------------|------|------|---------------------|------|------|-------------------------------|-----|------|---------------------|-----|------|-------------------|-----|------|-----|------|-----|
| Used Columns:                 |      |      |                            |     |     | Used Algorithms:  |      |      |                     |      |      | Used Columns:                 |     |     |                     |     |     | Used Algorithms:  |      |      |                     |      |      | Used Columns:                 |     |      |                     |     |      | Used Algorithms:  |     |      |     |      |     |
| Wkly plan created             |      |      | 14-40, 30-40, 30-40, 30-40 |     |     | Wkly plan created |      |      | 14-40, 30-40, 30-40 |      |      | Wkly plan created             |     |     | 14-40, 30-40, 30-40 |     |     | Wkly plan created |      |      | 14-40, 30-40, 30-40 |      |      | Wkly plan created             |     |      | 14-40, 30-40, 30-40 |     |      | Wkly plan created |     |      |     |      |     |
| Wkly plan created             |      |      | 14-40, 30-40, 30-40, 30-40 |     |     | Wkly plan created |      |      | 14-40, 30-40, 30-40 |      |      | Wkly plan created             |     |     | 14-40, 30-40, 30-40 |     |     | Wkly plan created |      |      | 14-40, 30-40, 30-40 |      |      | Wkly plan created             |     |      | 14-40, 30-40, 30-40 |     |      | Wkly plan created |     |      |     |      |     |
| S                             | GR   | 849  | 1101                       | 257 | 702 | 813               | 818  | 276  | 828                 | 1295 | 1295 | S                             | 730 | 815 | 801                 | 873 | 673 | 848               | 877  | 875  | 871                 | 875  | S    | 471                           | 886 | 479  | 815                 | 568 | 715  | 865               | 521 | 808  | 717 | 808  |     |
| S                             | 1001 | 1112 | 842                        | 914 | 587 | 1044              | 1046 | 1252 | 1503                | 925  | 925  | S                             | 585 | 565 | 1092                | 919 | 925 | 881               | 1054 | 1054 | 1054                | 1054 | 1054 | S                             | 711 | 1156 | 1151                | 885 | 1065 | 968               | 961 | 2579 | 822 | 1151 | 822 |
| S                             | 1002 | 1108 | 842                        | 914 | 587 | 1044              | 1046 | 1252 | 1503                | 925  | 925  | S                             | 586 | 566 | 1093                | 920 | 926 | 882               | 1055 | 1055 | 1055                | 1055 | 1055 | S                             | 712 | 1157 | 1152                | 886 | 1066 | 969               | 962 | 2579 | 823 | 1152 | 823 |
| S                             | 1003 | 1109 | 846                        | 916 | 587 | 1045              | 1047 | 1254 | 1504                | 926  | 926  | S                             | 587 | 567 | 1094                | 921 | 927 | 883               | 1056 | 1056 | 1056                | 1056 | 1056 | S                             | 713 | 1158 | 1153                | 887 | 1067 | 970               | 963 | 2579 | 824 | 1153 | 824 |
| S                             | 1004 | 1110 | 846                        | 916 | 587 | 1045              | 1047 | 1254 | 1504                | 926  | 926  | S                             | 588 | 568 | 1095                | 922 | 928 | 884               | 1057 | 1057 | 1057                | 1057 | 1057 | S                             | 714 | 1159 | 1154                | 888 | 1068 | 971               | 964 | 2579 | 825 | 1154 | 825 |
| S                             | 1005 | 1109 | 846                        | 916 | 587 | 1045              | 1047 | 1254 | 1504                | 926  | 926  | S                             | 589 | 569 | 1096                | 923 | 929 | 885               | 1058 | 1058 | 1058                | 1058 | 1058 | S                             | 715 | 1160 | 1155                | 889 | 1069 | 972               | 965 | 2579 | 826 | 1155 | 826 |
| S                             | 1006 | 1109 | 846                        | 916 | 587 | 1045              | 1047 | 1254 | 1504                | 926  | 926  | S                             | 590 | 569 | 1097                | 923 | 930 | 886               | 1059 | 1059 | 1059                | 1059 | 1059 | S                             | 716 | 1161 | 1156                | 890 | 1070 | 973               | 966 | 2579 | 827 | 1156 | 827 |
| S                             | 1007 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 591 | 570 | 1098                | 924 | 931 | 887               | 1060 | 1060 | 1060                | 1060 | 1060 | S                             | 717 | 1162 | 1157                | 891 | 1071 | 974               | 967 | 2579 | 828 | 1157 | 828 |
| S                             | 1008 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 592 | 571 | 1099                | 924 | 932 | 888               | 1061 | 1061 | 1061                | 1061 | 1061 | S                             | 718 | 1163 | 1158                | 892 | 1072 | 975               | 968 | 2579 | 829 | 1158 | 829 |
| S                             | 1009 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 593 | 571 | 1100                | 925 | 933 | 889               | 1062 | 1062 | 1062                | 1062 | 1062 | S                             | 719 | 1164 | 1159                | 893 | 1073 | 976               | 971 | 2579 | 830 | 1159 | 830 |
| S                             | 1010 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 594 | 572 | 1101                | 925 | 934 | 890               | 1063 | 1063 | 1063                | 1063 | 1063 | S                             | 720 | 1165 | 1160                | 894 | 1074 | 977               | 972 | 2579 | 831 | 1160 | 831 |
| S                             | 1011 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 595 | 572 | 1102                | 926 | 935 | 891               | 1064 | 1064 | 1064                | 1064 | 1064 | S                             | 721 | 1166 | 1161                | 895 | 1075 | 978               | 973 | 2579 | 832 | 1161 | 832 |
| S                             | 1012 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 596 | 573 | 1103                | 926 | 936 | 892               | 1065 | 1065 | 1065                | 1065 | 1065 | S                             | 722 | 1167 | 1162                | 896 | 1076 | 979               | 974 | 2579 | 833 | 1162 | 833 |
| S                             | 1013 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 597 | 573 | 1104                | 927 | 937 | 893               | 1066 | 1066 | 1066                | 1066 | 1066 | S                             | 723 | 1168 | 1163                | 897 | 1077 | 980               | 975 | 2579 | 834 | 1163 | 834 |
| S                             | 1014 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 598 | 574 | 1105                | 927 | 938 | 894               | 1067 | 1067 | 1067                | 1067 | 1067 | S                             | 724 | 1169 | 1164                | 898 | 1078 | 981               | 976 | 2579 | 835 | 1164 | 835 |
| S                             | 1015 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 599 | 574 | 1106                | 928 | 939 | 895               | 1068 | 1068 | 1068                | 1068 | 1068 | S                             | 725 | 1170 | 1165                | 899 | 1079 | 982               | 977 | 2579 | 836 | 1165 | 836 |
| S                             | 1016 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 600 | 575 | 1107                | 928 | 940 | 896               | 1069 | 1069 | 1069                | 1069 | 1069 | S                             | 726 | 1171 | 1166                | 900 | 1080 | 983               | 980 | 2579 | 837 | 1166 | 837 |
| S                             | 1017 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 601 | 575 | 1108                | 928 | 941 | 897               | 1070 | 1070 | 1070                | 1070 | 1070 | S                             | 727 | 1172 | 1167                | 901 | 1081 | 985               | 982 | 2579 | 838 | 1167 | 838 |
| S                             | 1018 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 602 | 576 | 1109                | 929 | 942 | 898               | 1071 | 1071 | 1071                | 1071 | 1071 | S                             | 728 | 1173 | 1168                | 902 | 1082 | 986               | 983 | 2579 | 839 | 1168 | 839 |
| S                             | 1019 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 603 | 576 | 1110                | 929 | 943 | 899               | 1072 | 1072 | 1072                | 1072 | 1072 | S                             | 729 | 1174 | 1169                | 903 | 1083 | 987               | 984 | 2579 | 840 | 1169 | 840 |
| S                             | 1020 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 604 | 577 | 1111                | 930 | 944 | 900               | 1073 | 1073 | 1073                | 1073 | 1073 | S                             | 730 | 1175 | 1170                | 904 | 1084 | 988               | 985 | 2579 | 841 | 1170 | 841 |
| S                             | 1021 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 605 | 577 | 1112                | 930 | 945 | 901               | 1074 | 1074 | 1074                | 1074 | 1074 | S                             | 731 | 1176 | 1171                | 905 | 1085 | 989               | 986 | 2579 | 842 | 1171 | 842 |
| S                             | 1022 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 606 | 578 | 1113                | 931 | 946 | 902               | 1075 | 1075 | 1075                | 1075 | 1075 | S                             | 732 | 1177 | 1172                | 906 | 1086 | 990               | 987 | 2579 | 843 | 1172 | 843 |
| S                             | 1023 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 607 | 578 | 1114                | 931 | 947 | 903               | 1076 | 1076 | 1076                | 1076 | 1076 | S                             | 733 | 1178 | 1173                | 907 | 1087 | 991               | 988 | 2579 | 844 | 1173 | 844 |
| S                             | 1024 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 608 | 579 | 1115                | 932 | 948 | 904               | 1077 | 1077 | 1077                | 1077 | 1077 | S                             | 734 | 1179 | 1174                | 908 | 1088 | 992               | 989 | 2579 | 845 | 1174 | 845 |
| S                             | 1025 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 609 | 579 | 1116                | 932 | 949 | 905               | 1078 | 1078 | 1078                | 1078 | 1078 | S                             | 735 | 1180 | 1175                | 909 | 1089 | 993               | 990 | 2579 | 846 | 1175 | 846 |
| S                             | 1026 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 610 | 579 | 1117                | 933 | 950 | 906               | 1079 | 1079 | 1079                | 1079 | 1079 | S                             | 736 | 1181 | 1176                | 910 | 1090 | 994               | 991 | 2579 | 847 | 1176 | 847 |
| S                             | 1027 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 611 | 579 | 1118                | 933 | 951 | 907               | 1080 | 1080 | 1080                | 1080 | 1080 | S                             | 737 | 1182 | 1177                | 911 | 1091 | 995               | 992 | 2579 | 848 | 1177 | 848 |
| S                             | 1028 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 612 | 579 | 1119                | 934 | 952 | 908               | 1081 | 1081 | 1081                | 1081 | 1081 | S                             | 738 | 1183 | 1178                | 912 | 1092 | 996               | 993 | 2579 | 849 | 1178 | 849 |
| S                             | 1029 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 613 | 579 | 1120                | 934 | 953 | 909               | 1082 | 1082 | 1082                | 1082 | 1082 | S                             | 739 | 1184 | 1183                | 913 | 1093 | 997               | 994 | 2579 | 850 | 1183 | 850 |
| S                             | 1030 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 614 | 579 | 1121                | 935 | 954 | 910               | 1083 | 1083 | 1083                | 1083 | 1083 | S                             | 740 | 1185 | 1184                | 914 | 1094 | 998               | 995 | 2579 | 851 | 1184 | 851 |
| S                             | 1031 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 615 | 579 | 1122                | 935 | 955 | 911               | 1084 | 1084 | 1084                | 1084 | 1084 | S                             | 741 | 1186 | 1185                | 915 | 1095 | 999               | 996 | 2579 | 852 | 1185 | 852 |
| S                             | 1032 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 616 | 579 | 1123                | 936 | 956 | 912               | 1085 | 1085 | 1085                | 1085 | 1085 | S                             | 742 | 1187 | 1186                | 916 | 1096 | 1000              | 997 | 2579 | 853 | 1186 | 853 |
| S                             | 1033 | 1107 | 847                        | 917 | 588 | 1046              | 1048 | 1255 | 1505                | 927  | 927  | S                             | 617 | 579 | 1124                | 936 | 957 | 913               | 1086 | 1086 | 1086                | 1086 | 1086 | S                             | 743 | 1188 | 1187                | 917 | 1097 | 1001              |     |      |     |      |     |

Figure 89 Structure of the initial data Excel files containing quantifications from image analysis. The technical replicates are represented by the three blocks of columns, while the three different endpoints are represented in different rows below each other. This representation might be comfortable for humans, but is not suitable for machine handling and parsing.

The information about the compounds and their tested concentrations have been stored in two separate Excel files. One provided information about which Excel file contained the raw data of which compound (Figure 90), and a second Excel file included the compound concentrations, plus some more general information about the compounds, such as the chemical formula, the molecular weight, the CAS No., the supplier, etc.

A)

| A  | B                               | C  | D                            | E                                  | F   | G                            | H           | I           | J           | K           | L           | M        |
|----|---------------------------------|--|------------------------------|------------------------------------|---|------------------------------|-------------|-------------|-------------|-------------|-------------|----------|
| 1  |                                 |  |                              |                                    |   |                              |             |             |             |             |             |          |
| 2  |                                 |  |                              |                                    |   |                              |             |             |             |             |             |          |
| 3  |                                 | name   | name                         | name                               | name  | name                         | if known    | if known |
| 4  | file name                       | biol repl substance 1 number (row B on plate)  | substance 2 (row C on plate) | substance 3 (row D on plate)       | substance 4 (row E on plate)                    | substance 5 (row F on plate) | substance 1 | substance 2 | substance 3 | substance 4 | substance 5 |          |
| 5  | 20140731_JD_Toxcast_sub01-05    | 1 Hydroxyurea  | Valinomycin                  | 3,3'-Imidopropionitril Thalidomide | 1-methyl-4-phenylpyrid                          | 127-07-1                     | 2001-95-8   | 111-94-4    | 50-35-1     | 36913-39-0  |             |          |
| 6  | 20140731_SG_Toxcast_sub01-05    | 2 Hydroxyurea  | Valinomycin                  | 3,3'-Imidopropionitril Thalidomide | 1-methyl-4-phenylpyrid                          | 127-07-1                     | 2001-95-8   | 111-94-4    | 50-35-1     | 36913-39-0  |             |          |
| 7  | 2n 20140815_SG_Toxcast_sub01-05 | 3 Hydroxyurea  | Valinomycin                  | 3,3'-Imidopropionitril Thalidomide | 1-methyl-4-phenylpyrid                          | 127-07-1                     | 2001-95-8   | 111-94-4    | 50-35-1     | 36913-39-0  |             |          |
| 8  | 20140731_JD_Toxcast_sub06-10    | 1 Saccharin Sodium salt h-6-Hydroxydopamine hyl Valproic acid sodium salt 2-Methoxyethanol |                              |                                    | Phenobarbital sodium s 2385-42-0                | 28094-15-7                   | 109-66-9    | 105-86-4    | 57-30-7     |             |             |          |
| 9  | 20140731_SG_Toxcast_sub06-10    | 2 Saccharin Sodium salt h-6-Hydroxydopamine hyl Valproic acid sodium salt 2-Methoxyethanol |                              |                                    | Phenobarbital sodium s 2385-42-0                | 28094-15-7                   | 109-66-9    | 105-86-4    | 57-30-7     |             |             |          |
| 10 | 2n 20140815_SG_Toxcast_sub06-10 | 3 Saccharin Sodium salt h-6-Hydroxydopamine hyl Valproic acid sodium salt 2-Methoxyethanol |                              |                                    | Phenobarbital sodium s 2385-42-0                | 28094-15-7                   | 109-66-9    | 105-86-4    | 57-30-7     |             |             |          |
| 11 | 20140731_JD_Toxcast_sub11-15    | 1,2- ethylhexyl diphenyl p-Dieldrin  | Pyrene                       |                                    | Tetraethylthiuram disul Di(2-ethylhexyl) phthal | 1241-94-7                    | 60-57-1     | 129-00-0    | 97-77-8     | 117-81-7    |             |          |
| 12 | 20140731_SG_Toxcast_sub11-15    | 2,2- ethylhexyl diphenyl p-Dieldrin  | Pyrene                       |                                    | Tetraethylthiuram disul Di(2-ethylhexyl) phthal | 1241-94-7                    | 60-57-1     | 129-00-0    | 97-77-8     | 117-81-7    |             |          |
| 13 | 2n 20140815_SG_Toxcast_sub11-15 | 3,2- ethylhexyl diphenyl p-Dieldrin  | Pyrene                       |                                    | Tetraethylthiuram disul Di(2-ethylhexyl) phthal | 1241-94-7                    | 60-57-1     | 129-00-0    | 97-77-8     | 117-81-7    |             |          |
| 42 | 20140731_JD_Toxcast_sub11-15    | 4 Bis(2-ethylhexyl) phthalate  | Pyrene                       |                                    | Phenobarbital sodium s 2385-42-0                | 28094-15-7                   | 109-66-9    | 105-86-4    | 57-30-7     |             |             |          |

B)

Figure 90 Structure of the Excel files containing information about the substances. The Excel file in A) links the substances to the according Excel files, while the Excel file in B) gives more information about the substances themselves (e.g. chemical formula, molecular weight, supplier, etc.).

Step 1: Re-structuring data to make them machine-friendly

In a first step, the data are extracted and re-arranged in a way that each row represents one data point, and technical replicates of the same data point (i.e. compound effect at a defined concentration) are not located in separate columns anymore. Having data from three plates in an actual plate-like layout next to each other in the same Excel file might be convenient for humans, but machines have a hard time parsing that structure. Using a loop, this is done for all 48 data files and the resulting data table is extended with each iteration. In the end all values (representing number of (viable) cells and neurite area) are now listed underneath each other in one column, while the endpoint they represent is indicated in another column , as well as the position on the cell plate and the technical replicate they originate from (Figure 91). This might not be that neat for humans anymore, but enables machines to easily parse the data and deal with it.

| S | Filename                     | D     | Value | S                  | Endpoint | S | CellPlate | S | CellPlate | S | TechnicalReplicate |
|---|------------------------------|-------|-------|--------------------|----------|---|-----------|---|-----------|---|--------------------|
|   | 20140731_JD_Toxcast_sub01-05 | 1,034 |       | Valid Object Count | B        |   | Col02     |   |           | 1 |                    |
|   | 20140731_JD_Toxcast_sub01-05 | 1,167 |       | Valid Object Count | B        |   | Col03     |   |           | 1 |                    |
|   | 20140731_JD_Toxcast_sub01-05 | 1,132 |       | Valid Object Count | B        |   | Col04     |   |           | 1 |                    |
|   | 20140731_JD_Toxcast_sub01-05 | 1,220 |       | Valid Object Count | B        |   | Col05     |   |           | 1 |                    |
|   | 20140731_JD_Toxcast_sub01-05 | 1,136 |       | Valid Object Count | B        |   | Col06     |   |           | 1 |                    |
|   | 20140731_JD_Toxcast_sub01-05 | 1,194 |       | Valid Object Count | B        |   | Col07     |   |           | 1 |                    |
|   | 20140731_JD_Toxcast_sub01-05 | 921   |       | Valid Object Count | B        |   | Col08     |   |           | 1 |                    |
|   | 20140731_JD_Toxcast_sub01-05 | 703   |       | Valid Object Count | B        |   | Col09     |   |           | 1 |                    |

Figure 91 Structure of the newly generated data table. Each data point is now represented by one row, while the endpoint is indicated in a separate column, as well as the technical replicate.

## Step 2: Adding the compound information

In a second step, the information about the tested compounds is added to the previously generated data table. Before, that information was stored in two separate Excel Tables. The data are re-arranged in a way that both compounds and CAS numbers are merged upon the file names of the original Excel input sheets. Information like supplier, structural formula, etc. are extracted and appended to the data table.

## Step 3: Enriching the data with chemical identifiers and controlled vocabulary

To improve interoperability and simplify data re-use in other contexts, the information about the chemicals has been extended by SMILES, InChIs and InChI keys. Originally the chemicals were provided only with CAS numbers as identifiers, but these are not very applicable for many purposes. We used REST API to retrieve the SMILES of the substances from an [NIH resource](#). These were then also converted to InChI and InChI keys using the RDKit KNIME Integration community nodes. This facilitates further use in a cheminformatic context, e.g. substructure search, etc.

Information about the biological target of a substance or its use is always nice to have. Therefore, we extend the metadata with information from the [ChEMBL](#) and [ChEBI](#) database using REST API. Knowing about the roles, effects and molecule type, for example, is not only useful for human researchers; having the identifiers and defined vocabulary of widely-used databases and ontologies in this field enhances interoperability and complies with the FAIR principles. Furthermore we extracted metadata about the request such as database and API version, release and query date.

## Step 4: Adding user-defined metadata

There are no intrinsic metadata to the quantified values available, but we want to add some user-defined metadata that will help others to understand the data table and give more information about each column and the data therein. To do so, we use the interactive Table Editor node which allows manual entries to an existing table. We added for example information about what the control samples actually are, units or links to the DB-ALM assay protocol and the ATCC cell line registry (DB-ALM is the EURL ECVAM Database on Alternative Methods to Animal Experimentation, ATCC is the American Type Culture Collection).

## Step 5: Writing the data and exporting the workflow summary

In the last step the data table and the metadata created in step 4 are exported as CSV files. We are using flow variables to ensure that the data table and the user-defined metadata share parts of the file name. This way the two files can be nicely matched. To ensure maximal transparency, we will also export the workflow summary of this workflow used to create the data table and metadata. This summary can be obtained either by 'File > Export > Workflow Summary (JSON/XML)...' menu in the KNIME Analytics Platform, or through a KNIME Server's REST API

(Figure 92). It can be either a JSON or XML file, and if you want to extract the most relevant information from that file, this component will help you.

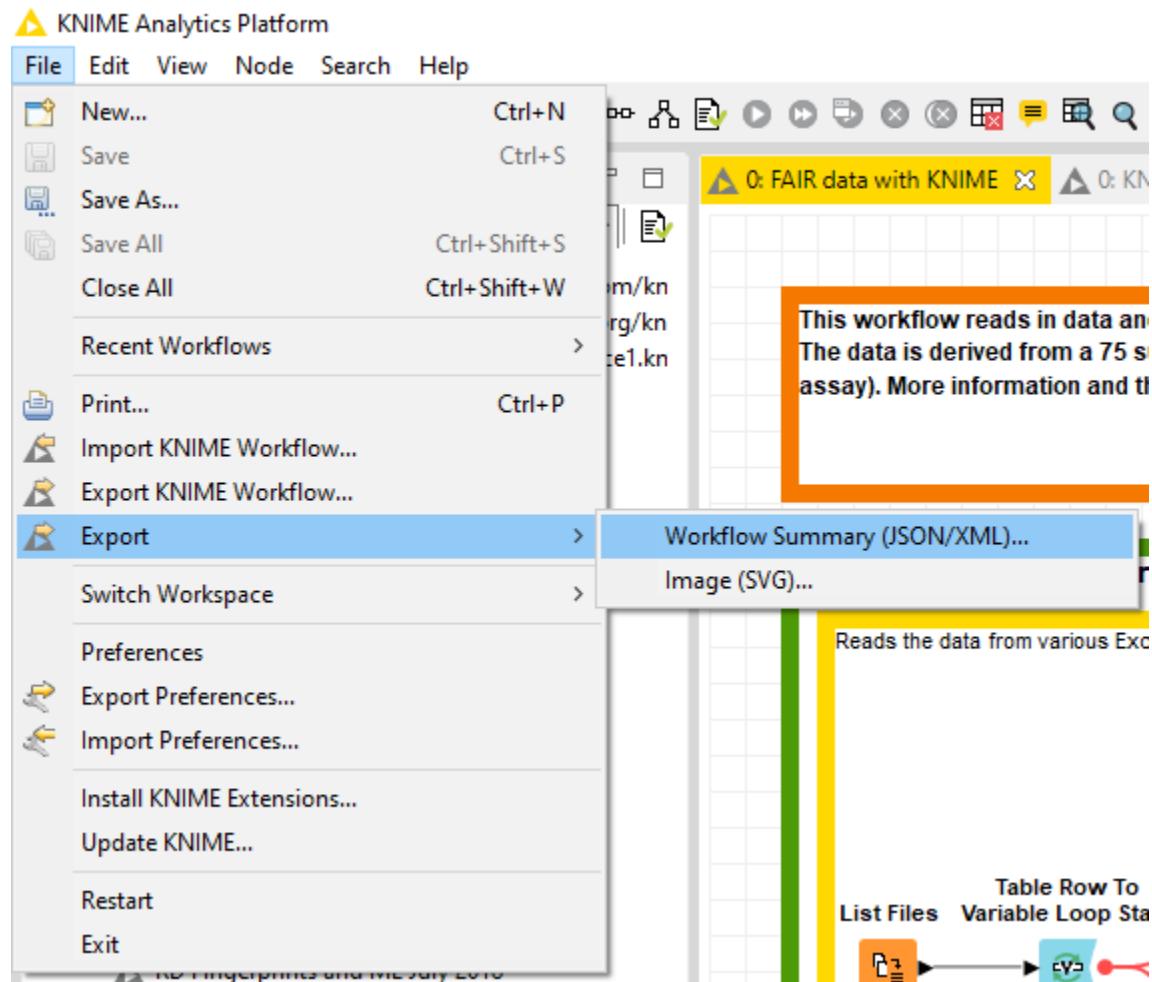


Figure 92 Exporting the workflow summary in XML or JSON format. The workflow summary contains information e.g. about the workflow version, the KNIME version it was created with, the operating system, the environment and information about all the nodes within the workflow.

It should be pointed out that by no means the use of KNIME alone makes data FAIR, and there are for sure aspects of the FAIR principles that KNIME is not suitable for (mostly concerning findability and accessibility), e.g. creating persistent, unique Identifiers (PID). The DOI (digital object identifier) is probably the most widely known and used persistent identifier, but these can only be assigned by DOI registration agencies (KNIME is not one of these). However, KNIME can be used to introduce identifiers of individual data points within the data set.

KNIME can contribute to **interoperability** and **reusability**, as we have shown in this blog post (Figure 93).

It can read, write and integrate numerous different file formats (serving principle I1), be used to extend the metadata with a controlled vocabulary (in this case GET requests to chEMBL and chEBI, serving principle I2)

Give qualified references (in this case extending the metadata with roles of the substances as defined in the chEBI ontology, principle I3)

It can add plenty of user-defined metadata e.g. the ID of the DB-ALM assay protocol, DOI to linked publications, the ATCC cell line registry or the contact details using the interactive Table Editor serves principle R1, and specifically R1.2.

The workflow summary provides information about how the data table was created, therefore also serving principle R1.

Any workflow itself will be assigned a unique short link when it's uploaded on the KNIME Hub, providing something identifier-like (we are well aware this is not a full-grown PID). Although the principle of accessibility is mostly covered by the data repositories, we want to mention that data can be shared within a workflow. As KNIME Analytics Platform is an open access data tool, this provides partners not only with the data, but also with the process with which it was created.

# Principle



## Findable



**F3.** Metadata clearly and explicitly include the identifier of the data they describe

**F4.** (Meta)data are registered or indexed in a searchable resource

Using flow variables to have (partially) the same file names for data tables and metadata

Write data to searchable resources and data repositories

## Interoperable



**I1.** (Meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.

**I2.** (Meta)data use vocabularies that follow FAIR principles

**I3.** (Meta)data include qualified references to other (meta)data

Reading and writing many different types of file formats, as well as connecting with databases

Adding information from discipline-specific databases and ontologies

Adding information retrieved from ontologies that contains qualified references

## Reusable



**R1.** (Meta)data are richly described with a plurality of accurate and relevant attributes

**R1.2.** (Meta)data are associated with detailed provenance

User-defined metadata and provenance can be added manually using the Interactive Table Editor. The KNIME workflow summary provides metadata about how the data set was created, and the workflow itself can be shared via the KNIME Hub

Figure 93 FAIR aspects that KNIME can contribute to.

## Summary of how KNIME contributed to FAIR data management

We have created a KNIME workflow that combines the data from 48 individual data Excel files into one large data table. While the Excel sheets were convenient for humans, they were not very machine-friendly. The data about the tested chemicals that were previously provided in two separate files are now integrated into the large data table. The chemical identifiers were extended by SMILES, InChI and InChI keys using the RDKit KNIME community nodes and REST API to enhance interoperability. The metadata was extended by domain-specific, controlled vocabulary using REST API and programmatic access to the chEMBL and chEBI databases. To comply with FAIRness, details about the used databases and ontologies are extracted as well. There is no 'intrinsic' metadata available e.g. for the image analysis, but we added plenty of 'user-defined' metadata. The data is now ready to be uploaded to a (project-specific) data repository, meeting the funding requirements of the project. Depending on the repository the (meta)data should be deposited at, KNIME also facilitates automatic upload using a PUT request. Project partners or

researchers working on follow-ups can now access a well-structured data table that enables them to recapitulate the published results and integrate it with newly generated data or data from chEMBL for example.

KNIME itself contributes to FAIR data management by providing extensive metadata about workflows (exportable workflow summaries), as well with the KNIME hub where workflows can be made publicly and persistently available by and for any user, ensuring transparency and reproducibility.

In case you are dealing with laboratory data such as microplate or imaging data, next week's blog post about retrieving raw data directly from laboratory devices and automatically analyzing it with KNIME Analytics Platform might be interesting for you as well. Go and check out the new SiLA2 integration in the Life Sciences space on the KNIME Hub.

## References

Delp J, Gutbier S, Klima S, et al. A high-throughput approach to identify specific neurotoxicants/developmental toxicants in human neuronal cell function assays [published correction appears in ALTEX. 2019;36(3):505]. ALTEX. 2018;35(2):235-253. doi:10.14573/altex.1712182

Krug AK, Balmer NV, Matt F, Schönenberger F, Merhof D, Leist M. Evaluation of a human neurite growth assay as specific screen for developmental neurotoxicants. Arch Toxicol. 2013;87(12):2215-2231. doi:10.1007/s00204-013-1072-y

Wilkinson MD, Dumontier M, Aalbersberg IJ, et al. The FAIR Guiding Principles for scientific data management and stewardship [published correction appears in Sci Data. 2019 Mar 19;6(1):6]. Sci Data. 2016;3:160018. Published 2016 Mar 15. doi:10.1038/sdata.2016.18

## All References

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., & Levine, A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12), 6745-6750.

<https://doi.org/10.1073/pnas.96.12.6745>

Barnes, R. J., Dhanoa, M. S., & Lister, S. J. (1989). Standard normal variate transformation and de-trending of near-infrared diffuse reflectance spectra. *Applied spectroscopy*, 43(5), 772-777.

<https://doi.org/10.1366/0003702894202201>

Callahan, B. J., McMurdie, P. J., Rosen, M. J., Han, A. W., Johnson, A. J. A., & Holmes, S. P. (2016). DADA2: high-resolution sample inference from Illumina amplicon data. *Nature methods*, 13(7), 581-583.

<https://doi.org/10.1038/nmeth.3869>

Cooper, G. M., & Shendure, J. (2011). Needles in stacks of needles: finding disease-causal variants in a wealth of genomic data. *Nature Reviews Genetics*, 12(9), 628-640. <https://doi.org/10.1038/nrg3046>

Danecek, P., Auton, A., Abecasis, G., Albers, C. A., Banks, E., DePristo, M. A., ... & Durbin, R. (2011). 1000 Genomes Project Analysis Group. The variant call format and vcftools. *Bioinformatics*, 27(15), 2156-2158. <https://doi.org/10.1093/bioinformatics/btr330>

Delp, J., Gutbier, S., Klima, S., Hoelting, L., Pinto-Gil, K., Hsieh, J. H., ... & Leist, M. (2018). A high-throughput approach to identify specific neurotoxicants/developmental toxicants in human neuronal cell function assays. *Alternatives to Animal Experimentation: ALTEX*, 35(2), 235-253.

<https://doi.org/10.14573/altex.1712182>

Distrutti, E., Monaldi, L., Ricci, P., & Fiorucci, S. (2016). Gut microbiota role in irritable bowel syndrome: New therapeutic strategies. *World journal of gastroenterology*, 22(7), 2219.

<https://doi.org/10.3748/wjg.v22.i7.2219>

GCCL Cardenas, R., D. Linhares, N., L. Ferreira, R., & Pena, S. D. (2017). Mendel, MD: a user-friendly open-source web tool for analyzing WES and WGS in the diagnosis of patients with Mendelian disorders. *PLoS computational biology*, 13(6), e1005520.

<https://doi.org/10.1371/journal.pcbi.1005520>

Halfvarson, J., Brislawn, C. J., Lamendella, R., Vázquez-Baeza, Y., Walters, W. A., Bramer, L. M., ... & Jansson, J. K. (2017). Dynamics of the human gut microbiome in inflammatory bowel disease. *Nature microbiology*, 2(5), 1-7. <https://doi.org/10.1038/nmicrobiol.2017.4>

Hastings, J., Owen, G., Dekker, A., Ennis, M., Kale, N., Muthukrishnan, V., ... & Steinbeck, C. (2016). ChEBI in 2016: Improved services and an expanding collection of metabolites. *Nucleic acids research*, 44(D1), D1214-D1219. <https://doi.org/10.1093/nar/gkv1031>

Hsin, C. H., Chou, Y. E., Yang, S. F., Su, S. C., Chuang, Y. T., Lin, S. H., & Lin, C. W. (2017). MMP-11 promoted the oral cancer migration and Fak/Src activation. *Oncotarget*, 8(20), 32783.

<https://doi.org/10.18632/oncotarget.15824>

- Karczewski, K. J., Francioli, L. C., Tiao, G., Cummings, B. B., Alföldi, J., Wang, Q., ... & MacArthur, D. G. (2020). The mutational constraint spectrum quantified from variation in 141,456 humans. *Nature*, 581(7809), 434-443. <https://doi.org/10.1038/s41586-020-2308-7>
- Kennedy, P. J., Cryan, J. F., Dinan, T. G., & Clarke, G. (2014). Irritable bowel syndrome: a microbiome-gut-brain axis disorder?. *World journal of gastroenterology: WJG*, 20(39), 14105. <https://doi.org/10.3748/wjg.v20.i39.14105>
- Khatri, P., Sirota, M., & Butte, A. J. (2012). Ten years of pathway analysis: current approaches and outstanding challenges. *PLoS Comput Biol*, 8(2), e1002375. <https://doi.org/10.1371/journal.pcbi.1002375>
- Krug, A. K., Balmer, N. V., Matt, F., Schönenberger, F., Merhof, D., & Leist, M. (2013). Evaluation of a human neurite growth assay as specific screen for developmental neurotoxicants. *Archives of toxicology*, 87(12), 2215-2231. <https://doi.org/10.1007/s00204-013-1072-y>
- Lyons, A. J., & Jones, J. (2007). Cell adhesion molecules, the extracellular matrix and oral squamous carcinoma. *International journal of oral and maxillofacial surgery*, 36(8), 671-679. <https://doi.org/10.1016/j.ijom.2007.04.002>
- McFarland, L. V., & Dublin, S. (2008). Meta-analysis of probiotics for the treatment of irritable bowel syndrome. *World journal of gastroenterology: WJG*, 14(17), 2650. <https://doi.org/10.3748/wjg.14.2650>
- McLaren, W., Gil, L., Hunt, S. E., Riat, H. S., Ritchie, G. R., Thormann, A., ... & Cunningham, F. (2016). The ensembl variant effect predictor. *Genome biology*, 17(1), 1-14. <https://doi.org/10.1186/s13059-016-0974-4>
- Ng, P. C., & Henikoff, S. (2003). SIFT: Predicting amino acid changes that affect protein function. *Nucleic acids research*, 31(13), 3812-3814. <https://doi.org/10.1093/nar/gkg509>
- Noble, W. S. (2009). How does multiple testing correction work?. *Nature biotechnology*, 27(12), 1135-1137. <https://doi.org/10.1038/nbt1209-1135>
- Porr, M., Lange, F., Marquard, D., Niemeyer, L., Lindner, P., Scheper, T., & Beutel, S. (2021). Implementing a digital infrastructure for the lab using a central laboratory server and the SiLA2 communication standard. *Engineering in life sciences*, 21(3-4), 208-219. <https://doi.org/10.1002/elsc.202000053>
- Pozuelo, M., Panda, S., Santiago, A., Mendez, S., Accarino, A., Santos, J., ... & Manichanh, C. (2015). Reduction of butyrate-and methane-producing microorganisms in patients with Irritable Bowel Syndrome. *Scientific reports*, 5(1), 1-12. <https://doi.org/10.1038/srep12693>
- Reimand, J., Isserlin, R., Voisin, V., Kucera, M., Tannus-Lopes, C., Rostamianfar, A., ... & Bader, G. D. (2019). Pathway enrichment analysis and visualization of omics data using g: Profiler, GSEA, Cytoscape and EnrichmentMap. *Nature protocols*, 14(2), 482-517. <https://doi.org/10.1038/s41596-018-0103-9>
- Robinson, M. D., McCarthy, D. J., & Smyth, G. K. (2010). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1), 139-140. <https://doi.org/10.1093/bioinformatics/btp616>

Ruairi Robertson,, 'Why the Gut Microbiome Is Crucial for Your Health', www.healthline.com, June 27, 2017, accessed Feb 2020, <https://www.healthline.com/nutrition/gut-microbiome-and-health>

Savitzky, A., & Golay, M. J. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36(8), 1627-1639. <https://doi.org/10.1021/ac60214a047>

Sefid Dashti, M. J., & Gamieldien, J. (2017). A practical guide to filtering and prioritizing genetic variants. *Biotechniques*, 62(1), 18-30. <https://www.future-science.com/doi/10.2144/000114492>

Stevens, A., & Ramirez-Lopez, L. (2014). An introduction to the prospectr package. *R Package Vignette, Report No.: R Package Version 0.1, 3.*

Sydow, D., Morger, A., Driller, M., & Volkamer, A. (2019). TeachOpenCADD: a teaching platform for computer-aided drug design using open source packages and data. *Journal of cheminformatics*, 11(1), 1-7. <https://doi.org/10.1186/s13321-019-0351-x>

Sydow, D., Wichmann, M., Rodríguez-Guerra, J., Goldmann, D., Landrum, G., & Volkamer, A. (2019). TeachOpenCADD-KNIME: a teaching platform for computer-aided drug design using KNIME workflows. *Journal of chemical information and modeling*, 59(10), 4083-4086. <https://doi.org/10.1021/acs.jcim.9b00662>

Tuch, B. B., Laborde, R. R., Xu, X., Gu, J., Chung, C. B., Monighetti, C. K., ... & Smith, D. I. (2010). Tumor transcriptome sequencing reveals allelic expression imbalances associated with copy number alterations. *PloS one*, 5(2), e9317. <https://doi.org/10.1371/journal.pone.0009317>

Wang, Y., & Qian, P. Y. (2009). Conservative fragments in bacterial 16S rRNA genes and primer design for 16S ribosomal DNA amplicons in metagenomic studies. *PloS one*, 4(10), e7401. <https://doi.org/10.1371/journal.pone.0007401>

Whetzel, P. L., Noy, N. F., Shah, N. H., Alexander, P. R., Nyulas, C., Tudorache, T., & Musen, M. A. (2011). BioPortal: enhanced functionality via new Web services from the National Center for Biomedical Ontology to access and use ontologies in software applications. *Nucleic acids research*, 39(suppl\_2), W541-W545. <https://doi.org/10.1093/nar/gkr469>

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., ... & Mons, B. (2016). The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data*, 3(1), 1-9. <https://doi.org/10.1038/sdata.2016.18>

Yang, H., Jiang, P., Liu, D., Wang, H. Q., Deng, Q., Niu, X., ... & Yang, W. (2019). Matrix metalloproteinase 11 is a potential therapeutic target in lung adenocarcinoma. *Molecular Therapy-Oncolytics*, 14, 82-93. <https://doi.org/10.1016/j.omto.2019.03.012>

Zhang, X., Huang, S., Guo, J., Zhou, L., You, L., Zhang, T., & Zhao, Y. (2016). Insights into the distinct roles of MMP-11 in tumor biology and future therapeutics. *International journal of oncology*, 48(5), 1783-1793. <https://doi.org/10.3892/ijo.2016.3400>

# Index

## A

|                                 |    |
|---------------------------------|----|
| ADME .....                      | 8  |
| ASV .....                       | 47 |
| ATC Classification System ..... | 69 |

## B

|                         |    |
|-------------------------|----|
| Balanced accuracy ..... | 18 |
| Bioinformatics .....    | 33 |

## C

|                                  |    |
|----------------------------------|----|
| ChEBI.....                       | 82 |
| CHEMBL .....                     | 7  |
| Chemistry Ontology.....          | 82 |
| Clustering.....                  | 39 |
| Cohen's kappa.....               | 18 |
| compound similarity.....         | 10 |
| Computer Aided Drug Design ..... | 5  |
| Corpus creation .....            | 69 |

## D

|                           |    |
|---------------------------|----|
| Dictionary creation ..... | 69 |
|---------------------------|----|

## E

|                                   |    |
|-----------------------------------|----|
| European Nucleotide Archive ..... | 44 |
|-----------------------------------|----|

## F

|                               |     |
|-------------------------------|-----|
| FAIR guiding principles ..... | 114 |
| FASTQ .....                   | 46  |
| FMCS algorithm.....           | 12  |
| F-measure.....                | 18  |

## G

|                            |    |
|----------------------------|----|
| Gene Expression Data ..... | 34 |
| Group compounds.....       | 11 |

## I

|   |    |
|---|----|
| IBD Patients .....                      | 45 |
| Integrated Deployment.....              | 19 |
| interactive bioactivity prediction..... | 25 |

## L

|                     |     |
|---------------------|-----|
| Lab Automation..... | 108 |
| Lab Data.....       | 92  |

## M

|                                  |     |
|----------------------------------|-----|
| maximum common substructure..... | 12  |
| Microbiomes.....                 | 44  |
| Microplate Data .....            | 111 |
| Model selection .....            | 15  |
| Multi-task neural network .....  | 25  |

## N

|  |    |
|--|----|
| Near Infrared Spectroscopy (NIR) ..... | 93 |
|--|----|

## O

|          |    |
|----------|----|
| OWL..... | 76 |
|----------|----|

## P

|                                  |    |
|----------------------------------|----|
| parameter optimization.....      | 15 |
| Pathway enrichment .....         | 40 |
| PCA analysis.....                | 99 |
| PDB .....                        | 14 |
| Preprocessed Spectral Data ..... | 99 |
| Purpose of a Drug .....          | 67 |

## R

|                                |    |
|--------------------------------|----|
| RDF.....                       | 76 |
| Reactome Pathway Database..... | 40 |

## S

|  |     |
|--|-----|
| Screen compounds using machine learning..... | 13  |
| Semantic Web .....                           | 76  |
| SiLA integration .....                       | 108 |
| SMILES.....                                  | 118 |
| SPARQL.....                                  | 84  |

## T

|                         |    |
|-------------------------|----|
| taxonomic profile ..... | 51 |
| Text Mining .....       | 66 |

|                                      |                              |    |
|--------------------------------------|------------------------------|----|
| <b>U</b>                             | Variant Prioritization ..... | 57 |
| unwanted substructures .....         | 9                            |    |
| <b>V</b>                             |                              |    |
| Variant Effect Predictor (VEP) ..... | 61                           |    |
| <b>W</b>                             |                              |    |
| web application .....                | 21                           |    |

*About the Author:*

**Alice Krebs**

Alice is a data scientist in the Life Sciences team at KNIME. She has a PhD from the University of Konstanz, where she worked in in vitro toxicology, test development, and risk assessment. Her strong interest is in gaining insight from data in scientific areas such as cheminformatics, toxicology, drug development and clinical research.