

Edited by
DAYANJAN S WIJESINGHE & ALI ASGHAR MARVI

Boosting Digital Health

Data-driven Applications for Digital Healthcare

KNIME v5.2



Copyright © 2024 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 has been updated for **KNIME 5.2**.

For information regarding permissions and sales, write to:

KNIME Press

Talacker 50

8001 Zurich

Switzerland

knimepress@knime.com

www.knime.com

Preface

In an era where healthcare is evolving at an unprecedented pace, the integration of digital technologies for optimized and efficient care delivery is no longer just a futuristic vision, but a tangible reality. The realm of Digital Health applications stands at the forefront of this transformation, offering innovative solutions to improve healthcare delivery. This book delves into the heart of this transformation, exploring how healthcare providers, traditionally distant from the intricacies of computer programming, are now pivotal in harnessing the power of digital health technologies.

Healthcare providers, with their in-depth understanding of patient needs and healthcare dynamics, are best positioned to identify where Digital Health applications can be most effective. However, the development of these applications has historically required a skill set in computer programming – a proficiency not typically included in professional healthcare training programs. This discrepancy often leaves those with the most relevant insights without the tools to implement them.

Enter KNIME, a game-changer in the field when applied to Digital Health. KNIME revolutionizes the creation of data and analysis workflows, in a codeless fashion through an intuitive interface. This has now made it possible to train future healthcare professionals in developing Digital Health applications without necessitating a background in computer programming.

The success of this approach is nothing short of remarkable: We have witnessed healthcare professional students create comprehensive digital health workflows using KNIME within a mere five weeks. This book serves as a testament to this success, showcasing how the low/no-code KNIME platform has been effectively harnessed by professional healthcare degree students to create digital health applications relevant to real-world healthcare practice.

Through a series of compelling case studies and real-life applications, this book demonstrates the transformative power of integrating digital technology into healthcare. It is a tribute to the innovative use of KNIME in empowering healthcare professionals and a guide for future pioneers in the field of Digital Health.

Welcome to a journey of discovery, innovation, and digital transformation in healthcare.

Dayanjan S. Wijesinghe

Associate Professor

Virginia Commonwealth University, School of Pharmacy

Table of Contents

<u>DATA APPS FOR HEALTHCARE APPLICATIONS</u>	1
HOW A DATA APP IMPROVES VANCOMYCIN DOSING IN OBESITY	3
AUTOMATING TOTAL PARENTERAL NUTRITION CALCULATION IN KNIME FOR QUALITY CARE	11
MONITOR KIDNEY HEALTH IN KNIME	15
AUTOMATING PHARMACOKINETICS CALCULATION IN KNIME	21
HEPARIN THERAPEUTIC MONITORING AND CALCULATION WITH KNIME	27
TRACK DISEASE WITH KNIME ON COVID-19 DASHBOARD	32
<u>ANALYZING DIGITAL HEALTHCARE DATA</u>	35
INTERACT WITH EPIC ON FHIR TO VISUALIZE PATIENT DATA	36
HOW TO USE FAERS FOR ADVERSE REACTION MANAGEMENT	42
USING KNIME TO UNDERTAKE PHARMACOVIGILANCE OF SPECIAL PATIENT POPULATIONS	46
<u>DEEP LEARNING FOR DIGITAL HEALTHCARE</u>	51
PREDICT BLOOD GLUCOSE WITH AN LSTM ON CGM DATA	52
LEARN HOW TO CLASSIFY ELECTROCARDIOGRAM SIGNALS WITH DEEP LEARNING IN KNIME	59
HOW TO PERFORM ELECTROCARDIOGRAM CATEGORIZATION AND DETECT ARRHYTHMIA	67
<u>LEVERAGING HEALTHCARE LITERATURE</u>	72
TAGGING DISEASE NAMES IN BIOMEDICAL LITERATURE	73
IMPROVE LITERATURE SEARCH & MINIMIZE INFORMATION OVERLOAD	81
<u>NODE & TOPIC INDEX</u>	88

Data Apps for Healthcare Applications

Digital health(care) is a discipline in which the involved parties have different backgrounds and unequal degrees of data and analytics literacy: Solution developers and patients are on opposite ends of this spectrum. Typically, clinicians and practitioners (like many of the authors of articles in this book) lie in between these two extremes with deep domain knowledge, the demand to provide the best care to patients, and usually little to no coding experience.

KNIME Analytics Platform enables practitioners to implement advanced calculations and personalized analyses with significant ease. Its no-code/low-code approach to visual workflows grants access to a wide variety of features without the necessity to code (which often turns out to be a hurdle) but leaves the option open. In addition, KNIME Business Hub enables users to deploy visual workflows as data apps with a few clicks. This allows for easy accessibility and execution from any browser, making insights available to anyone, even patients.

Since workflows can easily be shared through the KNIME Community Hub, anyone can use available workflows as blueprints for new projects. The workflows associated with articles in this chapter (and the rest of the book) are collected in a live repository on KNIME Community Hub and can be downloaded and customized. They share the common focus of encapsulating advanced calculations in simple workflows that can easily be used by practitioners.

This chapter includes the articles:

- **How a Data App improves Vancomycin Dosing in Obesity**, p. 3
 - Dayanjan Wijesinghe, *Virginia Commonwealth University* &
 - Danielle Holdren, *Virginia Commonwealth University*
- **Automating Total Parenteral Nutrition Calculation in KNIME for Quality Care**, p. 11
 - Dayanjan Wijesinghe, *Virginia Commonwealth University* &
 - Courtney Ciarrocca, *Virginia Commonwealth University*
- **Monitor Kidney Health in KNIME**, p. 15
 - Mallik Graves, *Virginia Commonwealth University* &
 - Dayanjan Wijesinghe, *Virginia Commonwealth University*

- **Automating Pharmacokinetics Calculation in KNIME**, p. 21
 - Danielle Holdren, *Virginia Commonwealth University* &
 - Dayanjan Wijesinghe, *Virginia Commonwealth University*
- **Heparin Therapeutic Monitoring and Calculation with KNIME**, p. 27
 - Amir Behdani, *Virginia Commonwealth University*
 - Micah Buller, *Virginia Commonwealth University*
 - Gina Chong, *Virginia Commonwealth University*
 - Maria DePonte, *Virginia Commonwealth University*
 - ReHanshae Harvey, *Virginia Commonwealth University*
 - Sebastian Jaques, *Virginia Commonwealth University*
 - Dori Leka, *Virginia Commonwealth University* &
 - Dayanjan Wijesinghe, *Virginia Commonwealth University*
- **Track Disease with KNIME on COVID-19 Dashboard**, p. 32
 - Ali Asghar Marvi, *KNIME*

How a Data App improves Vancomycin Dosing in Obesity

Authors: [Dayanjan Wijesinghe](#), Virginia Commonwealth University & [Danielle Holdren](#), Virginia Commonwealth University

Workflow on KNIME Community Hub: [Vancomycin Input Calculator](#)

Timely Vancomycin Area-Under-the-Curve-based Dosing for Informed Decision-making

[Vancomycin](#) is an antibiotic used to treat serious bacterial infections. It's most commonly administered intravenously, typically for infections such as bacteremia, endocarditis, osteomyelitis, etc. Despite its wide-spread use and success in treating these serious infections, clinicians have a challenging job to provide the correct dosage to their patients. This is because the manner in which Vancomycin distributes through the body is complex, with factors such as body weight, muscle mass, and fat distribution all playing a role. Vancomycin also has a narrow therapeutic index. This means that small differences in dose or blood concentration can lead to therapeutic failures or adverse drug reactions and an increased risk of overdosing or underdosing.

Vancomycin dosing guidelines therefore recommend close adjustments based on total-body-weight. However recent research shows that this is not ideal for obese patients. While the volume of distribution of vancomycin increases with body weight, it does not increase proportionally. The results of incorrect dosing of Vancomycin are serious. Overdosing can lead to liver toxicity, hearing loss, and blood clots, for example, and underdosing to prolonged hospitalization, even treatment failure.

A new approach to dosing Vancomycin for obese patient populations using so-called AUC-guided monitoring method produces more accurate dosing. However, these calculations are often time consuming and can be a major hindrance in scenarios where time to antibiotic administration is critical. Due to the recent nature of this newly recommended dosing approach, there is a lack of automated tools available for use in clinical practice at this time.

In this article we look at how we were able to overcome the challenges of calculating Vancomycin dosing for patients with obesity and build an automated Vancomycin dosage calculator. The calculator incorporates the customized AUC-guided requirements, gives clinicians access to precise dosage recommendations, and the ability to respond quickly to insight from [Therapeutic Drug Monitoring \(TDM\)](#).

The Importance of Therapeutic Drug Monitoring with Vancomycin

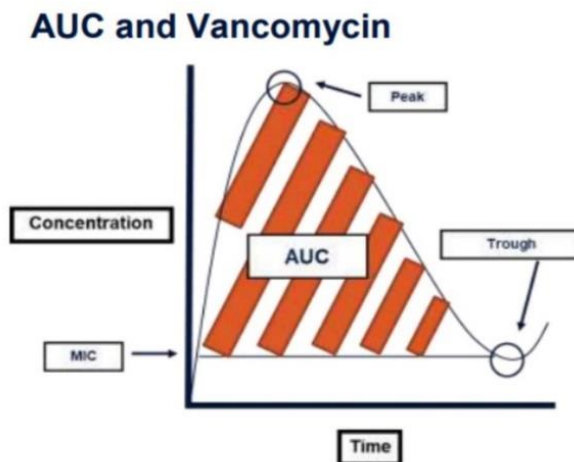
Vancomycin's narrow therapeutic index requires that Vancomycin dosing is monitored according to Therapeutic Drug Monitoring (TDM) and dosings adjusted quickly and accurately.

TDM allows clinicians to measure and track drug levels in the patient's body throughout the course of treatment. By tracking these levels, clinicians are better able to dose Vancomycin within its therapeutic index, and reduce the risks of over- or underdosing.

This monitoring of drug levels involves the collection of blood samples at specified times during treatment, followed by laboratory testing, and the use of pharmacokinetic-pharmacodynamic (PKPD) calculations to determine the true Vancomycin concentration in the blood – what the drug is currently “doing” to the body.

Area-Under-the-Curve (AUC) Monitoring

The most common PKPD parameters used to predict the concentration of Vancomycin in the blood include the [Minimum Inhibitory Concentration \(MIC\)](#), peak and trough levels, and the area-under-the-curve (AUC) (see figure below¹).



Area under the curve illustration¹.

¹ Adult Vancomycin AUC-Guided Dosing Playbook. HCA Healthcare. Created March 1, 2022. Accessed July 13, 2022.

The MIC is the lowest concentration of antibiotic which prevents visible growth of bacteria. This concentration is dependent on the patient, the antibiotic, and the offending bacterial pathogen, which may vary across different clinical scenarios. The MIC for vancomycin is assumed to be 1 mg/L in most clinical scenarios.

The peak level is the highest concentration of antibiotic reached after administration, whereas the trough level is the lowest concentration of antibiotic reached.

The AUC refers to the total exposure of antibiotic over a specified period of time, also known as the dosing interval. Certain antibiotics, such as vancomycin, are considered to be time dependent, and demonstrate the best bacterial inhibition when the total antibiotic exposure remains above the target MIC.

Therefore, the best predictor of Vancomycin activity in the body is an AUC/MIC ratio and can be used to guide dosing decisions.

Challenges in Calculating Vancomycin

Previous vancomycin guidelines recommended using trough-guided monitoring of vancomycin, in which trough levels were used as a surrogate measure of the patient's AUC². This method suggested that a trough level of >10 mg/L would be sufficient to sustain an AUC goal of >400 mg*h/L, which is the specific AUC at which vancomycin is determined to be effective.

However, further research has determined that trough levels have poor correlation with the AUC, often resulting in underestimation and increased antibiotic exposure.

Recent updates to the vancomycin guidelines place a strong emphasis on the use of AUC-guided monitoring in patients with severe MRSA infections, who will be receiving >5 days of therapy, and may be at increased risk of liver toxicity³. This monitoring method involves the collection of peak and trough levels, followed by laboratory testing, and PKPD calculations of the patient's true AUC. For these patients, an AUC of 400-600 mg*h/L should be targeted.

² Rybak MJ et al. *Vancomycin Therapeutic Guidelines: A Summary of Consensus Recommendations from the Infectious Diseases Society of America, the American Society of Health-System Pharmacists, and the Society of Infectious Diseases Pharmacists. Clinical Infectious Diseases. 2009; 49(3): 325–327.*
<https://doi.org/10.1086/600877>.

³ Rybak MJ et al. *Therapeutic Monitoring of Vancomycin for Serious Methicillin-resistant Staphylococcus aureus Infections: A Revised Consensus Guideline and Review by the American Society of Health-system Pharmacists, the Infectious Diseases Society of America, the Pediatric Infectious Diseases Society, and the Society of Infectious Diseases Pharmacists. Clinical Infectious Diseases. 2020; 71(6): 1361–1364.*
<https://doi.org/10.1093/cid/ciaa303>.

New Approach to Vancomycin Calculation Helps Obese Patient Populations

Recent research from Masich et. al.⁴, further expands upon this recommendation as it relates to the patient population with obesity, where certain PKPD parameters may be altered from the standard patient population. Studies indicate that while the volume of distribution of vancomycin does increase with body weight, it does not increase proportionally.

In patients with obesity, the distribution of vancomycin in the body may increase or decrease depending on patient specific factors, such as body fat content, muscle mass, and kidney function. In her research, Masich develops a novel approach to dosing vancomycin in this patient population using the newly emphasized AUC-guided monitoring method.

However, these calculations can often be time consuming in the clinical setting and can be a hindrance in scenarios where time to antibiotic administration is critical. The transition to AUC-guided monitoring adds steps and calculations that may not have been previously relied upon. Additionally, due to the recent nature of this recommendation, there is a lack of automated tools available for use in clinical practice at this time. Standard IT approaches to automating this type of calculation are simply not suited to the individualized nature of Vancomycin dosing for patients with obesity.

The use of an automated tool for vancomycin calculations would overcome several barriers in clinical practice: time constraints, dosing inaccuracies, and special population considerations.

How a Customized, Automated Tool Enables Precise Treatment

The complex nature of vancomycin dosing often requires a reliance upon healthcare professionals with an in-depth knowledge of PKPD parameters. A code free data analytics platform, such as KNIME Analytics Platform⁵, can be used to build a workflow that makes the calculations with the necessary customization. Advanced data science tools provide the flexibility to build such a customized and complex application. KNIME's visual programming environment bridges the coding skills gap. Healthcare

⁴ Masich AM et al. *Vancomycin Pharmacokinetics in Obese Patients with Sepsis or Septic Shock*. *Pharmacotherapy*. 2020; 40(3): 211-220. doi:10.1002/phar.2367.

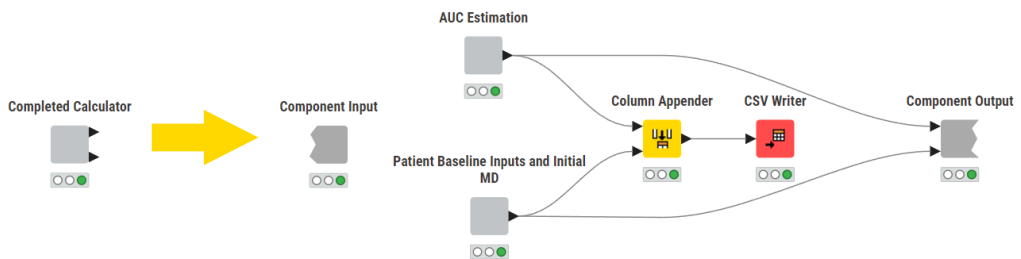
⁵ Berthold, M. R. et al. *KNIME: The Konstanz Information Miner. Data Analysis, Machine Learning and Applications: Proceedings of the 31st Annual Conference of the gesellschaft für Klassifikation E.V., Albert-Ludwigs-Universität Freiburg*. 2007; 319–326. https://doi.org/10.1007/978-3-540-78246-9_38.

professionals can build analytics applications without needing to know how to code and are able to fold their valuable expertise into the process.

The Vancomycin Calculator in KNIME

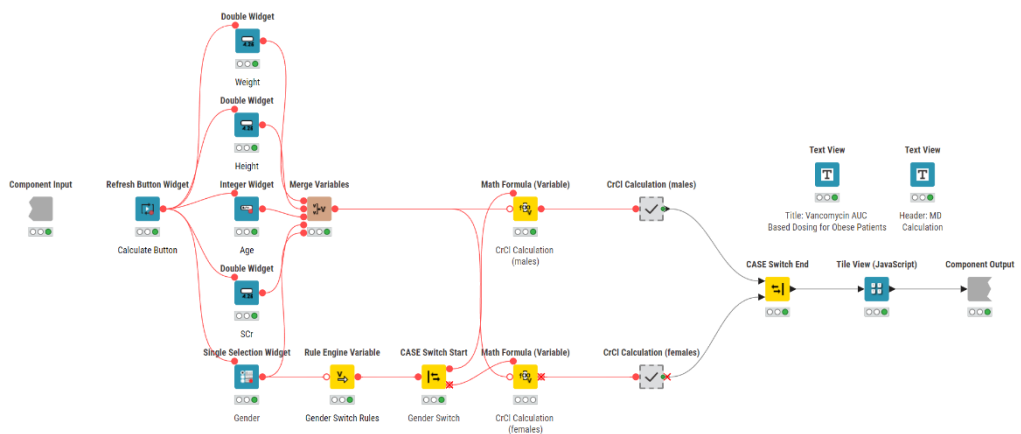
Over the span of a five-week advanced pharmacy practice experience (APPE) in digital health, a PharmD candidate with no prior knowledge of data analytics or coding, has successfully created an interactive vancomycin AUC-based dosing calculator for use in patients with obesity.

The underlying workflow comprises a series of components, which encapsulate the different calculation routes necessary for the patient's final AUC estimation.



Start of calculator workflow leading to PKPD calculations.

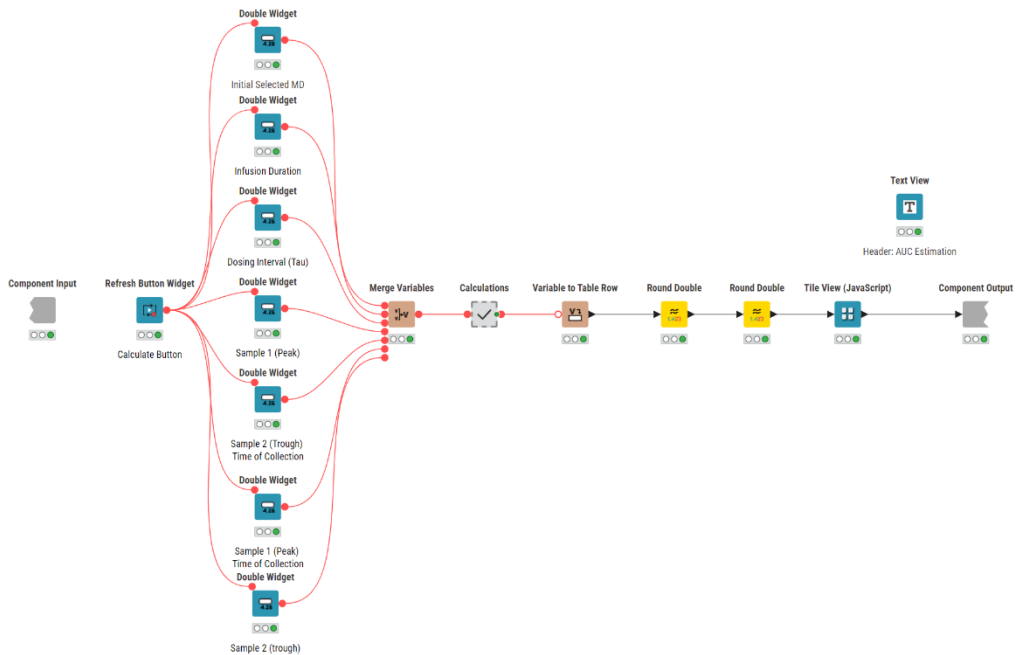
Moving through the workflow, you'll see a component that collects a patient's baseline inputs and calculates an initial maintenance dose (MD) of vancomycin (*Patient Baseline Inputs and Initial MD*). This component also allows for the calculation of other individualized data, such as body mass index (BMI) and creatinine clearance (CrCl). The *Widget* nodes and *Tile View* node included here begin to build the interactive dashboard (see figure below).



Workflow of the component used to calculate the patient's initial MD of vancomycin.

Data Apps for Healthcare Applications
How a Data App improves Vancomycin Dosing in Obesity

The next component of the workflow handles the first order PKPD calculations (*AUC Estimation*). This component encapsulates many different equations essential for estimating the biggest output of interest, which is the patient's AUC concentration. In a similar fashion, the *Widget* nodes and *Tile View* nodes add functionality to the interactive dashboard (see figure below).



Workflow of the component used to perform first order PKPD calculations for the patient's AUC estimation.

Each of the two components, *Patient Baseline Inputs* and *Initial MD* and *AUC Estimation*, serve as the building blocks for the interactive calculator dashboard, which provides the functionality to individualize dosing for each patient (see figure below).

Vancomycin AUC Based Dosing Calculator for Obese Patients

Maintenance Dose Calculation

Gender Selection

Female

Age

40

Height

155

Weight (kg)

80

Serum Creatinine

1,5

Calculate

CrCl: 63
Weight (kg): 80
Serum Creatinine (SCr): 2
Age: 40
Height (cm): 155
BMI: 52
Initial Maintenance Dose Calculation (lower range): 1767
Initial Maintenance Dose Calculation (higher range): 2650

Showing 1 to 1 of 1 entries

AUC Estimation

Initial Selected MD (mg)

4500

Dosing Interval (hrs)

12

Infusion Duration (hrs)

1

Sample 1 (Peak)

30

Sample 2 (Trough)

15

Sample 1 (Peak) Collection Time

2

Sample 2 (Trough) Collection Time

10

Calculate

AUC Estimation: 487
Clearance (Cl): 18
Volume of Distribution (Vd): 204
True Trough: 13
True Peak: 33
Elimination Rate Constant (k): 0.087
Selected Maintenance Dose (mg): 4500
Sample 2 (Trough): 15
Sample 1 (Peak): 30
Dosing Interval (hrs): 12
Infusion Duration (hrs): 1

Showing 1 to 1 of 1 entries

Interactive calculator dashboard which can be shared as a browser-based data app.

AUC-based Dosing Calculation Reduced to Seconds

The developed KNIME workflow captures each PKPD calculation step used in the estimation of a patient's AUC, and appropriately outputs values which can be used to guide clinical decision making. These calculations, which are usually time consuming, are condensed to seconds. This allows clinicians to visualize different data inputs in a more efficient and agile manner, simplifying the vancomycin dosing process in clinical practice.

Agile Vancomycin Dosing for Better Patient Outcomes

By optimizing dosing based on the calculator, we are going to be able to prevent toxicities but also prevent the overtreatment that can happen if we don't reach the therapeutic levels. Ultimately this will result in improved patient outcomes, decreased hospitalization length, and increased time to recovery.

Watch the presentation on YouTube: ["Danielle Holdren - Vancomycin Area Under the Curve-Based Dosing in Obesity: Calculation Dashboard"](#).

Special thanks to Alexander Smart PharmD, BCIDP and Anne Masich PharmD, BCPS for their support and feedback throughout.

Automating Total Parenteral Nutrition Calculation in KNIME for Quality Care

Authors: [Dayanjan Wijesinghe](#), Virginia Commonwealth University & [Courtney Ciarrocca](#), Virginia Commonwealth University

Workflow on KNIME Community Hub: [TPN Calculator](#)

Imagine that you are the clinical pharmacist in the internal medicine team at your hospital. During morning rounds, you see a new patient who was admitted that morning for an 8-day history of nausea, vomiting, abdominal pain and fever. The physician taking care of the new patient tells you they have tried to place a feeding tube in his nose, but he keeps pulling it out. Until they find a diagnosis for his N/V the team wants to start a TPN and asks for your help.

Patients hospitalized for serious illnesses or injury are often unable to obtain their daily nutrition needs orally. Lengthy treatments can mean that patients with gastrointestinal absorption issues, obstructions, or persistent hemorrhages, for example, are not allowed anything by mouth for several days, and more.

Total Parenteral Nutrition, also known as a **TPN**, provides patients with their daily nutritional needs via their bloodstream. TPNs are vital in healthcare because they provide life-saving nutrition for people with intestinal failure. These patients would otherwise be classified “unfeedable” as they would have no means to obtain the nutrients their body needs.

The body undergoes immense stress during critical illness which can lead to multiple organ dysfunction, longer hospital stays, and a disproportionate death rate. Coupled with the increased caloric deficit that arises when no nutrition is available, trials have shown that these complications are significantly worsened for patients who do not receive nutritional support. They bring metabolic balance back to the patient as each component is personalized for the individual. Maintaining this proper nutrition in critically ill patients reduces the body’s response to stress, prevents cellular injury, and helps balance the immune response, all of which leads to better health outcomes.

However, the calculation of TPNs is complicated. TPNs comprise not only the variety of proteins, carbohydrates, lipids, and electrolytes the body needs to function properly. They can also be further personalized by adding multivitamins, trace elements, and medications, such as insulin and those needed to prevent stress ulcers.

The “Moving Parts” of TPN Calculation

But it's the many moving parts in the equation that make parenteral nutrition calculation even more complicated.

Multiple patient factors influence the final mix of ingredients that go into the “TPN order”: The form the healthcare professional fills out to specify what the patient needs based on their status.

Any Change Has Multiple Downstream Effects

Any small change in the patient's status will alter the final make-up of their TPN order. Any small change in the products used for TPNs can mean a change in ratio of the required nutrients. This impacts the required quantity needed and subsequently the downstream calculations.

For example, electrolyte balances need to be checked to ensure that the proper acid-base balance is achieved. Osmolarity also needs to be checked to determine which IV route can be safely used for administration. It is also important to not overfeed the patient. Many of the patients who receive TPNs are malnourished, and overfeeding can increase stress on vital organs and cause serious health problems.

TPN calculations are a balancing act between getting the patient what they need and not overloading them. Precise monitoring is required to make daily changes to the TPN, adjusting to the changing lab values.

How KNIME Simplifies TPN Calculations

With so many variables affecting TPN calculation, there is considerable potential for error. Automating TPN calculations increases efficiency and minimizes error. However, due to the changing nature of a patient's dietary requirements, available products etc., automating these calculations using standard IT-driven approaches is challenging. Conventional tools might not be flexible enough for such a customized solution. In addition, the important concepts of TPN calculation have to be explained to the IT team – to non-healthcare specialists – in order for them to be able to provide an automated solution.

This process can be made more efficient when the healthcare specialists are able to build their own automated solution.

5 Weeks from Project Start to Finish: TPN Calculator in KNIME

During the final part of my *Advanced Pharmacy Practice Elective (APPE)* rotation, we used KNIME to build an automated solution. As a no-code/low-code data science tool, it gave me – as a healthcare specialist with no coding background – ease of learning and access to advanced analytics.

The final product of this rotation uses KNIME Analytics Platform for each aspect of the calculation and creates an interactive dashboard – all without a single line of code. In combination with the KNIME Business Hub, this dashboard can be shared or made centrally available as a browser-based [data app](#).

Have a look at the result.

Ciarrocca-Wijesinghe TPN Calculator
* for student use *

Patient Sex

☒ Male ☐ Female

Patient Age (years)

65

Weight (kg)

67

Patient Height (cm)

168

Labs	Patient Value	Units
Na	143	mEq/L
K	3.7	mEq/L
Cl	99	mEq/L
Ca	7	mg/dL
Mg	1.9	mg/dL
Phos	2.6	mg/dL
CO2	24	mEq/L
Albumin	2.4	g/dL

Calculate

Does Calcium Need to be Corrected? (Is Albumin < 4?)

☒ Yes ☐ No

Fluid Requirements Calculation Method

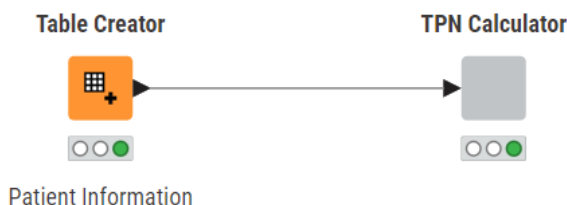
Holliday-Segar Method

Fluids from Other Sources (mL/day)

250

Calculate

The TPN calculator dashboard.



The workflow that creates the TPN calculator dashboard.

Doing the calculations by hand can be quite complicated in order to ensure everything is balanced. By using the TPN calculator workflow, the patient and product data can be entered and the final TPN order calculated in just a few seconds.

The interactivity of the calculator also allows users to quickly see how changing a part of the order will affect the final product. It also alerts the pharmacist to when ratios go

over or below thresholds. This solution could be advanced further in KNIME by automating patient data loading for a central hospital database.

"At the School of Pharmacy at VCU, we see the project as a proof of concept that demonstrates how someone who is educated in a healthcare field and understands TPNs but knows very little about coding, can really quickly implement a workflow of relevance to healthcare practice."

Within the span of a five-week *Advanced Pharmacy Practice Elective (APPE)* rotation, someone with no knowledge of KNIME or coding has successfully managed to create a tool to simplify TPN calculations.

Watch the presentation on YouTube: ["Courtney Ciarrocca - Digital Health APPE Rotation - Total Parenteral Nutrition Calculation Dashboard"](#).

Monitor Kidney Health in KNIME

Authors: [Dayanjan Wijesinghe](#), Virginia Commonwealth University & [Malik Graves](#), Virginia Commonwealth University

Workflow on KNIME Community Hub: [Creatinine Clearance, Glomerular Filtration Rate, Drug Dosage Calculation Dashboard](#)

Kidney health is key to our overall wellbeing. Healthcare professionals use a measurement known as the Glomerular Filtration Rate (GFR) to monitor how well our kidneys are working to clean our blood.

Glomeruli are tiny filters in our kidneys that help remove toxins from our blood. The GFR measures how much blood these filters can clean every minute. Monitoring our GFR helps doctors keep an eye out for the onset of kidney disease – the major disease of concern being Chronic Kidney Disease (CDK) – and provide the right degree of individualized treatment, ranging from advice on simple lifestyle changes through to kidney transplant or dialysis.

However, it can't be measured directly. The calculation is complex, based on multiple factors like age, body size, sex, race/ethnicity, as well as the level of creatinine – a waste product – in the blood.

Pharmacists strive to avoid the risks associated with the progression of kidney disease, of overdosing or underdosing – indeed any factors that potentially hinder optimal delivery of treatments and the patient's recovery. Accurate calculation of GFR and Creatinine Clearance (CrCl) are therefore vital to minimizing risks to patients.

The Challenges of Measuring CrCl and GFR

The GFR reveals the rate at which fluid is filtered by the kidneys, while CrCl, measure how much creatinine is in your blood. Healthy kidneys filter it out, but if you have kidney disease, creatinine stays in the blood and gradually builds up.

Creatinine levels and GFR are both measured by using creatinine samples from serum or urine. Creatinine collected from one source can be compared with creatinine from the other source. For comparison purposes, serum collection has to be within 24 hours of urine collection and vice versa. Serum creatinine samples are typically collected more than urine samples.

Healthcare professionals need the CrCl calculations in order to provide the estimated GFR – as GFR is not measured directly in practice. But despite the fact that CrCl

calculations are performed commonly, GFR calculation is not error free. CrCl overestimates GFR by 10-20%. This is due to creatinine being freely filtered by the glomerulus while also being secreted by the capillaries.

Further challenges to measuring creatinine clearance involve improper urine collection. This leads to an underestimation of creatinine excretion. Age-related increases in tubular secretion result in overestimation of GFR.

Other challenges include sex and race, with specific factors affecting creatinine production and clearance.

The Challenges of Calculating Drug Dosages

The difficulty here is that healthcare professionals need to complete a thorough examination of the patient's history. Knowing and understanding the patient's chief complaint, the history of the current illness, their prior medical history, social history, family history, medication, and prior treatments is all crucial to deciding which types of medication can be prescribed and at what dose.

To sum it up, every patient is different. There is no "one size fits all" treatment option. Fortunately, there are practice guidelines that can be referenced, but these still have to be tailored to best suit the patient. Many factors play a part. One important factor, for example, outside of guideline recommendations and best practices, is patient preference. Healthcare professionals need to be aware that just because a reliable source suggests a particular method this does not mean it will always be successful.

What Do the Numbers Mean?

High serum creatinine levels and low creatinine clearance indicate abnormal renal (kidney) function.

A normal creatinine clearance level for healthy women is 100-130mL/min or 110-150 mL/min for men. People whose kidneys are functioning normally will have a serum creatinine level 0.5 and 1.1mg/dl for women or 0.6 to 1.2 mg/dl for men.

How are the Numbers Calculated?

As we already mentioned, CrCl is measured by monitoring creatinine levels. We use the [Cockcroft-Gault](#) formula⁶, which includes a patient's weight (kg), age and gender, to estimate CrCl in mg/dL.

⁶ https://www.kidney.org/professionals/kdoqi/gfr_calculatorcoc

For GFR, there are several formulas you can use:

- Modification of Diet in Renal Disease Study Group (MDRD) – includes serum creatinine, age, ethnicity, and albumin levels. Recommended for detecting a GFR lower than 60 mL per minute or lower than 90 mL in older patients.
- Chronic Kidney Disease Epidemiology Collaboration (CKD-EPI) – based on gender and whether a patient is black or non-black. Practice guidelines recommend this formula for the estimation of GFR.
- Mayo Quadratic formula – used to better estimate GFR in patients that have preserved renal function
- Schwartz formula – estimation of GFR in children

Free Up Valuable Time with Automated, On-the-Fly Calculations

Easily accessible automation is one of the main reasons why we think KNIME is a good platform for Creatinine Clearance, Glomerular Filtration Rate and Drug Dosage Calculations.

I have used KNIME to build calculators that can be used in a clinical setting. They have the potential to free up a healthcare professional's time – bypassing the need to look up appropriate calculation formulas, nephrotoxic medications, and dosing guidelines.

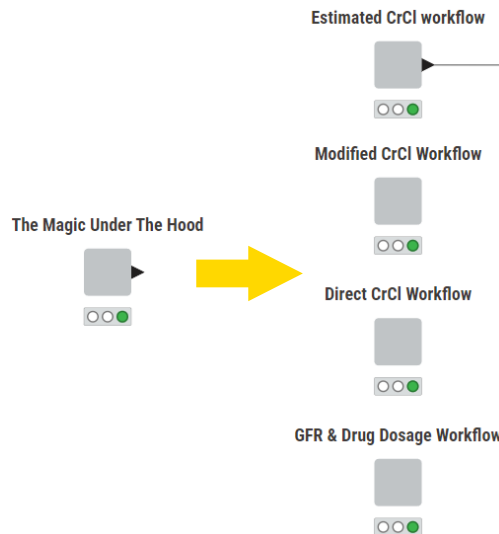
With my interactive calculator everything the healthcare professional needs is available in one place. It provides the means to calculate the gold standard calculation of creatinine clearance, gives you the normal ranges for men and women. enabling you to refer and check your calculations.

The calculator also adds in the ability to make your calculations more precise with two further methods of CrCl calculation depending on whether the patient is at home or hospitalized.

"Take it from me, a fourth-year pharmacy student with minimal programming knowledge is now a third level certified KNIME user - It took 1-2 weeks to gain my certification in using KNIME; within 3 weeks I was comfortable with application practices. I have learned KNIME concepts and utilization for my five-week advanced pharmacy practice experience (APPE) digital health rotation. KNIME provides its users with many tools, and even has the capability to incorporate outside functionality when the integrated tools are not enough. The community too is steadily growing, broadening the possibilities of what KNIME can do, across various practice settings." – Malik Graves

The Magic Under the Calculator's Hood: The KNIME Workflow

My workflow consists of four main components to calculate Estimated CrCl, Modified CrCl, Direct CrCl, GFR & Drug Dosage. Each component is then made up of many different nodes and widgets that coexist to create the completed interactive calculators, accessible via the KNIME Business Hub.



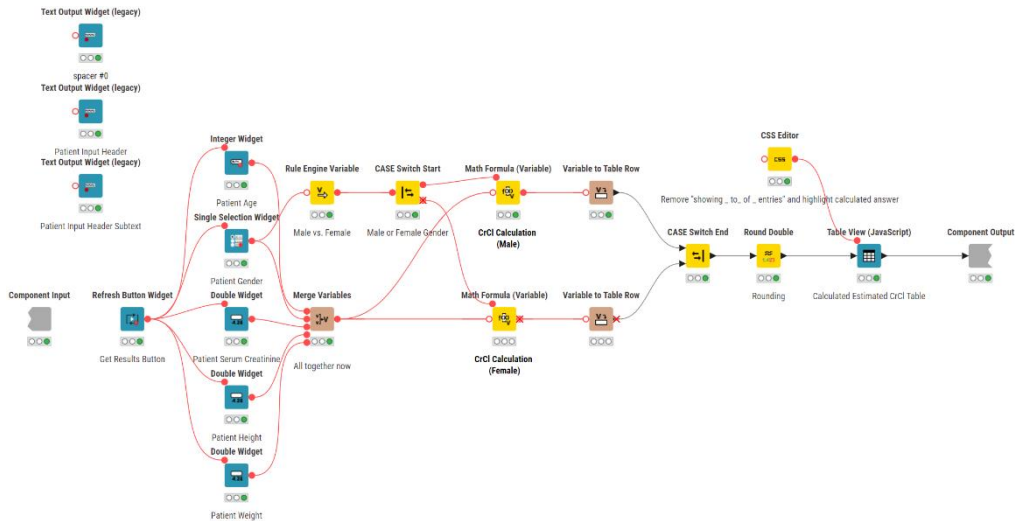
Workflow broken down into four main components.

Below you will find an example of what the inside of each component looks like, give or take.

Widget nodes (blue) create how the interactive part of the calculator will look, *Math Formula* nodes carry out the calculations and numbering aspects (yellow), and *Flow Variable* nodes mark what data is to be used (brown).

Data Apps for Healthcare Applications

Monitor Kidney Health in KNIME



The inside one the Estimated CrCl workflow components. The Math formula nodes carry out the calculations.

Interactive Data App for Estimated CrCl Calculation

When everything is placed together as desired, complete with instructions, you have an interactive tool to utilize at your command.

Input Your Values Here For An Estimated CrCl Calculation

Use this method for a general calculation (gold standard)
Utilizes the Cockcroft-Gault equation
$$(140 - \text{age}) \times \text{Weight (kg)} \times [0.85 \text{ if female}] / 72 \times [\text{Serum Creatinine (mg/dL)}]$$

Patient Age [years]

0

Patient Gender

Male

Patient Weight [kilograms]

0

Patient Height [inches]

0

Patient Serum Creatinine (SCr) [mg/dL]

0

Get Results!

Estimated CrCl (mL/min)

NaN

Interactive CrCl Calculator. Input patient values to calculate.

Data Apps for Healthcare Applications

Monitor Kidney Health in KNIME

Stages of CKD	eGFR ranges	Explanation
G1 (Stage 1)	≥90	Normal or High
G2 (Stage 2)	60-90	Mildly decreased
G3a (Stage 3)	45-59	Mildly to moderately decreased
G3b (Stage 3)	30-44	Moderately to severely decreased
G4 (Stage 4)	15-29	Severely decreased
G5 (Stage 5)	<15	Kidney failure

Built In Drug Dosage Calculator

*Depending on kidney function (GFR value), dose adjustments are necessary when prescribing certain medications.
Choose from a list of common medications below that require renal dosage adjustment
***Medications are color coded: ACE inhibitors = Red, Beta blocker = Blue, Thiazide diuretic = Yellow
****Once the desired medication is selected, click on the Get Results! button to receive a recommendation.
*****Doses provided are to be used as a reference only. Only take doses prescribed by your doctor.

Select medication:

Benazepril

Get Results!

Calculated GFR (mL/min/m ²)	Stage	Drug Name	Recommended Dosage
255	G1 (Stage 1)	Benazepril	Take 10 mg by mouth daily

Interactive Drug Dosage Calculator. Input patient values to calculate.

5 Weeks from Project Start to Project Finish

My short-term goal was, as the first digital health student to have L3 certification, to have my completed interactive calculator available in the digital healthcare repository on the KNIME Business Hub for others to use. Check!

Next Steps: Connect Data App to Lexicomp Drug Database

My next steps now are to make updated versions of the calculator, adding more input options, to work towards achieving my overall desire to connect the calculator to Lexicomp's drug database. This achievement would allow vast selection and appropriate dosing of many medications, far more than what is available now.

Automating Pharmacokinetics Calculation in KNIME

Authors: [Dayanjan Wijesinghe](#), Virginia Commonwealth University & [Danielle Holdren](#), Virginia Commonwealth University

Workflow on KNIME Community Hub: [PK Calculator](#)

Using an Automated Tool in Clinical Practice for Efficient PK Calculations

Pharmacokinetics is a branch of pharmacology that provides insight on how the body responds to a drug. It encompasses processes such as **A**bsorption, **D**istribution, **M**etabolism, and **E**xcretion (ADME), which can be used to determine how efficient and how safe a drug is. Clinicians use pharmacokinetic parameters and calculations to visualize and interpret each of the phases of ADME as they monitor a drug's action in vivo.

Each phase of ADME is associated with specific pharmacokinetic parameters. For example, the absorption and distribution phases of ADME can be extrapolated from drug concentrations and volumes, while the metabolism and excretion phases may be illustrated by parameters such as the elimination rate constant and clearance. Other considerations include the route and method of drug administration, which also plays a role in the body's response to a drug. In practice, each of these pharmacokinetic parameters can be calculated, and a treatment approach can be individualized for patients.

However, one of the challenges in pharmacokinetics (PK) is the time and computational power these calculations require.

Using an automated tool in clinical practice can provide a more efficient process of drug monitoring via pharmacokinetic parameters. I used the code free data analytics platform, KNIME, to create a customized workflow capable of performing these pharmacokinetic calculations. KNIME is a platform which provides users, who may have minimum knowledge of coding and analytics, with the necessary tools to build such a calculator.

A KNIME Workflow for Pharmacokinetic Calculation

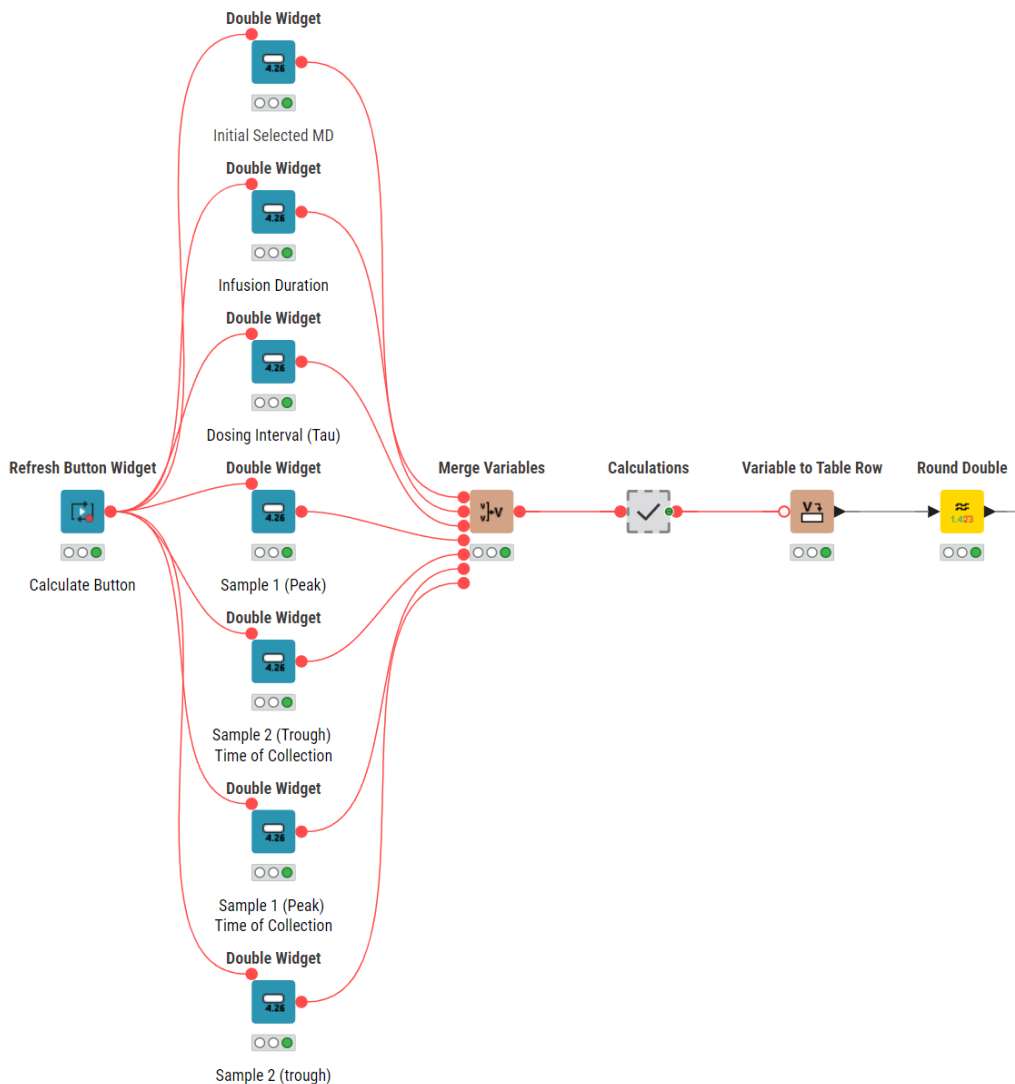
Pharmacokinetic calculation differs depending on how the drug is administered. My pharmacokinetic workflow consists of a single component, which contains different calculations routes of calculations, necessary for estimating different pharmacokinetic parameters.

The workflow is split into three main routes. In the figure below, you can see the pharmacokinetic calculations necessary for drugs administered intravenously (IV).

IV + PO Dosing



Start of calculator workflow leading to pharmacokinetic calculations.



Workflow snippet of pharmacokinetic calculations for IV administered drugs.

Here you can see how this part of the workflow looks when viewed via the interactive dashboard.

Dosing Inputs

Initial Selected MD (mg)

1000,8

Sample 2 (Trough)

8,9

Dosing Interval (hrs)

12

Sample 1 (Peak) Collection Time

2

Infusion Duration (hrs)

1

Sample 2 (Trough) Collection Time

10

Sample 1 (Peak)

30

Calculate

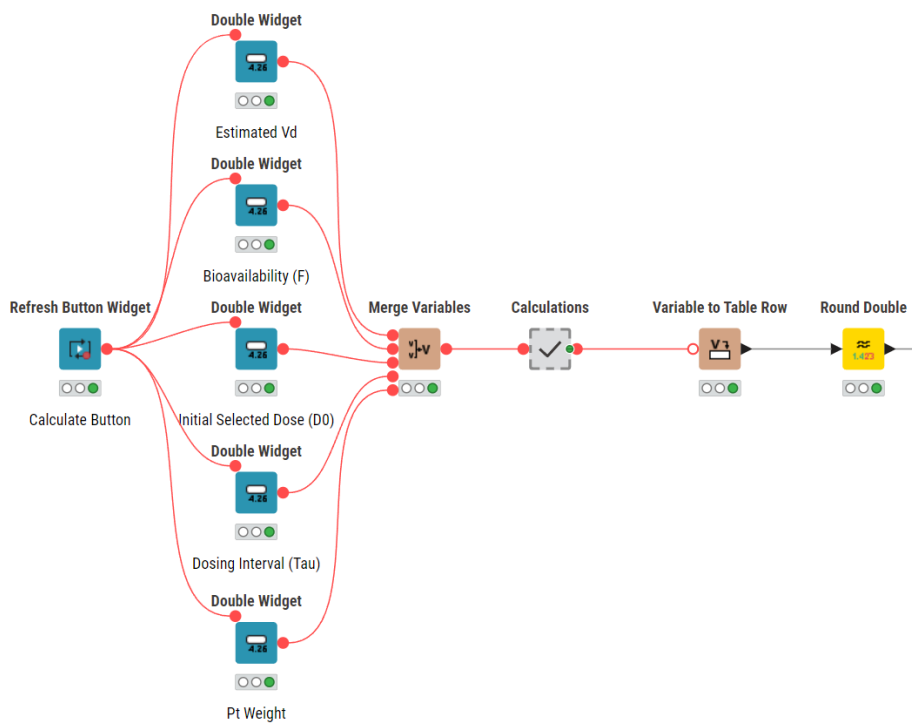
Elimination Rate Constant (k)	0.152
Half-life (hr)	5
Volume of Distribution (L)	32
Clearance (hr)	5
Initial Concentration (mg/ml)	32
Steady State Concentration (mg/ml)	17
True Peak (mg/ml)	35
True Trough (mg/ml)	7
AUC Estimation (mg*hr/L)	394

Showing 1 to 9 of 9 entries

Entering the information required for pharmacokinetic calculations based on IV administered drugs.

The figure below shows the calculations necessary for drugs administered orally (PO).

Data Apps for Healthcare Applications
Automating Pharmacokinetics Calculation in KNIME



Workflow snippet of pharmacokinetic calculations for PO administered drugs.

This workflow snippet below underlies the part of the dashboard for entering and making the calculations based on orally administered drugs.

Oral (PO) Dosing

Initial Selected Dose (mg)
100

Enter Oral Bioavailability (F)
0,5

Patient Weight (kg)
65

Estimated Volume of Distribution (L/kg)
1,5

Dosing Interval (hrs)
12

Calculate

Clearance (L/hr)	0
Css Average (mg/L)	33
Estimated Volume of Distribution (L/kg)	1.5

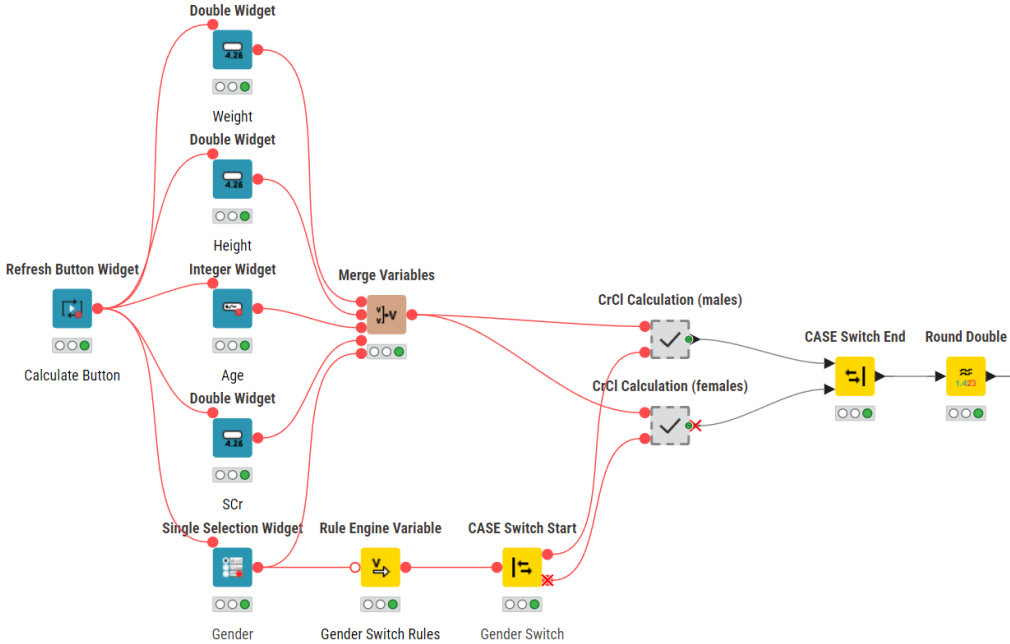
Showing 1 to 3 of 3 entries

AUC Estimation (mg*hr/L)	400
Maintenance Dose (mg)	100
Loading Dose (mg)	100

Showing 1 to 3 of 3 entries

Pharmacokinetic calculations based on orally administered drugs.

The calculation route, shown below, takes care of a series of baseline calculations useful for dosing purposes.



Workflow snippet of patient baseline calculations

The blue *Widget* nodes and *Table View* nodes included in this workflow are the foundation of the interactive dashboard, allowing for patient baseline inputs and visualization of calculated data. The yellow *Math Formula (Variable)* nodes allow for the embedding of necessary formulas for the pharmacokinetic calculations.

Below you can see how this part of the workflow looks when viewed via the interactive data app.

Patient Baseline Characteristics

Age (yrs)

30

Serum Creatinine (mg/dL)

1

Height (cm)

150

Gender Selection

Male

Weight (kg)

65

Calculate

IBW (kg)	BMI	CrCl (ml/min)
44	43	99

Showing 1 to 1 of 1 entries

Entering patient data to begin making the pharmacokinetic calculations.

Digital Health Application Development

This project was part of an *Advanced Pharmacy Practice Elective (APPE)* rotation on Digital Health Application Development at Virginia Commonwealth University, School of Pharmacy. We used KNIME to build an automated solution. As a no-code/low-code data science tool, it gave us – as a healthcare specialist with no coding background – ease of learning and access to advanced analytics.

Watch the presentation on YouTube: "[Danielle Holdren - Digital Health APPE Rotation - Pharmacokinetic Calculation Dashboard](#)".

Heparin Therapeutic Monitoring and Calculation with KNIME

Authors: [Dayanjan Wijesinghe](#), [Amir Behdani](#), [Micah Buller](#), [Gina Chong](#), [Maria DePonte](#), [ReHanshae Harvey](#), [Sebastian Jaques](#), & [Dori Leka](#) - Virginia Commonwealth University

Workflows on KNIME Community Hub: [Heparin Calculator for Adult General Cardiovascular and Adult and Children's Deep Vein Thrombosis and Peripheral Edema](#)

Heparin is an anticoagulant agent widely used in inpatient settings to prevent the formation of blood clots which can develop into deep vein thrombosis (DVT) and pulmonary embolism (PE). The risk for development of DVT and PE is highest in the hospital due to long-term bed rest. Signs and symptoms of DVT include unilateral leg swelling, pain, warmth, redness, palpable cord, and Homan's sign; whereas, signs and symptoms of PE include cough, chest pain/tightness, shortness of breath, palpitations, hemoptysis, tachypnea, tachycardia, diaphoresis, and extension of neck pain. Heparin dosing is very complicated. The provider has to calculate the initial bolus dose and infusion dose based on the patient's weight until the maximum dose is reached. From there a patient needs to be monitored closely using the Activated Partial Thromboplastin Time (aPTT) tool to see if a heparin dose adjustment is needed to prevent an adverse event from occurring. For instance, if a supra-therapeutic dose of heparin and sub-therapeutic dose of heparin is given it can cause bleeding and induce clots forming, respectively.

The Importance of Heparin Therapeutic Monitoring

All hospitals have a standardized heparin nomogram that utilizes the Activated Partial Thromboplastin Time (aPTT) to guide heparin dosing to reduce delays in therapy and adverse events. The aPTT is a measure of how long it takes for an individual to form a blood clot. With that being said, there are different heparin nomograms created for specific patient indications. For the sake of simplicity, we focused on creating a clinical calculator based on VCU's heparin dosing protocol for the adult general cardiovascular (CV), adult DVT/PE, and pediatric DVT/PE.

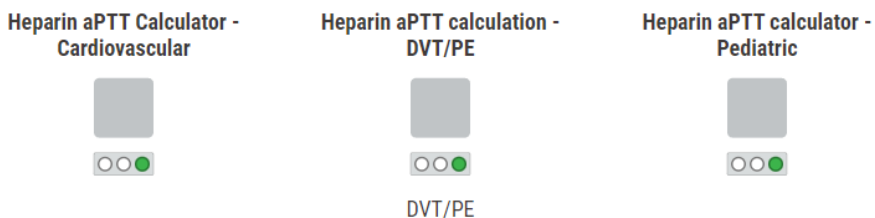
Therefore, a patient on heparin needs to be monitored closely using the Activated Partial Thromboplastin Time (aPPT).

Use of KNIME in Heparin Calculations

We utilized a free data analytics software called [KNIME](#) to create our simple input-output heparin dosing calculator. Our clinical calculator provides the initial bolus dose, initial infusion rate, infusion rate/bolus dose adjustment and time to repeat aPTT measurement for adult general CV, adult DVT/PE and pediatric DVT/PE cases. The goal of developing a heparin dosing calculator is to help ease the burden and stress for the healthcare provider, to reduce dosing errors, and to retrieve a dosing regimen rapidly.

Advantages of our KNIME Workflow in Comparison to our Competitors

There are other clinical calculators available for heparin dosing available online, such as GlobalRPh, Dr. Zad, and Heart Recovery. Unlike our competitors, our calculator is not product specific, available for different patient populations, and offers multiple heparin dosing protocols. We also use [an open source free platform, KNIME](#), which is available for anyone to use. Our competitors' clinical calculators are most likely written in code, which cannot be easily followed or understood by an ordinary person. This poses another advantage of our calculator since we utilize an abstract version of code called nodes. Lastly, our internal workflow is easily accessible to anyone who is interested in learning how we were able to create our calculator, and in return, can be effortlessly modified and applied as a starting point for others to use for their own projects.



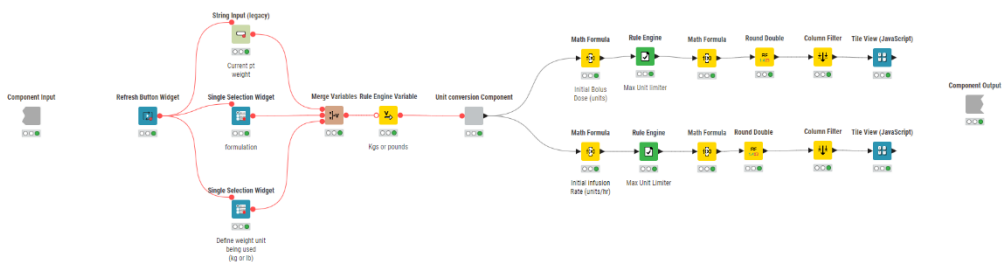
This figure depicts the three main sections of our KNIME workflow.

KNIME Workflow

Our workflow consists of three main sections, adult general CV, adult DVT/PE, and pediatric DVT/PE heparin dosing. Each section is broken into two separate components, one for initial bolus and initial infusion rate and the other for adjusted heparin dosing based on the patient's aPTT.

1. Adult general CV and DVT/PE

We will use the adult general CV calculator as an example. The first component in the workflow (figure below) is the interactive board that utilizes the *String Input* node and two *Single Selection Widget* nodes. These nodes allow the user to input patient specific information such as weight and the heparin concentration. Whether the weight is entered in pounds or kilograms, our workflow can convert the value into kilograms to use in the calculation of the initial doses through the *Rule Engine Variable* node and the *Unit conversion* component. The final portion handles heparin calculations to determine the initial bolus dose and initial infusion rate by using the *Math Formula*, *Rule Engine*, *Round Double*, and *Column Filter* nodes.



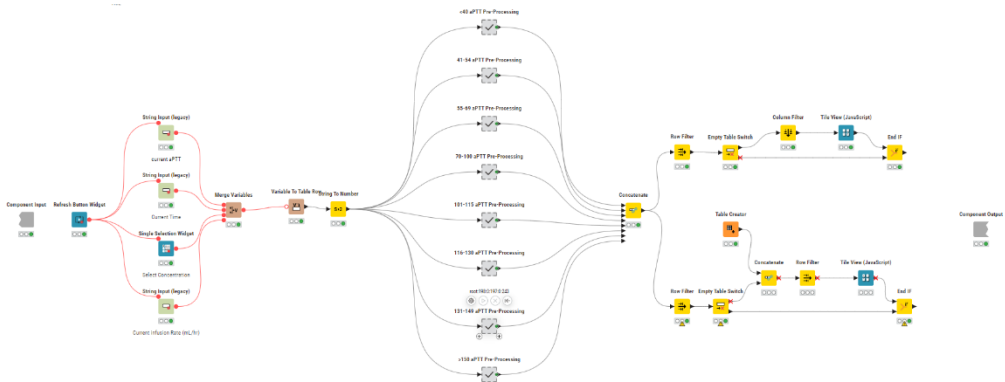
This figure depicts the workflow used to calculate the heparin initial bolus dose and initial infusion rate for adults.

The next component of the workflow, as shown in figure below, calculates the post-aPTT dose adjustments. It also allows the user to input the time, current aPTT, concentration of the heparin product used, and current infusion rate which are all merged and sorted based on aPTT. There are 8 different pathways the calculator will follow based on the aPTT entered in seconds: <40, 40–54, 55–69, 70–100, 101–115, 116–130, 131–150, and >150. With a therapeutic aPTT goal between 70–100 seconds. Each pathway will have a different outcome that will provide an adjusted bolus dose, if needed, and notify the provider whether they should stop the infusion, change the rate, and what time to take another measurement.

For adult DVT/PE dosing, the components are exactly the same. However, there are a couple of differences including the initial bolus and initial infusion values used in the calculations and the aPTT ranges that determine the calculator output.

Data Apps for Healthcare Applications

Heparin Therapeutic Monitoring and Calculation with KNIME



This figure is an overview of the workflow to determine the post-aPTT dose adjustments for adult heparin dosing.

Please use this calculator to adjust heparin infusion AFTER initial aPTT has been obtained.

Enter the current time in military time

Enter your measured aPTT (seconds)

Enter the current infusion rate (mL/hr)

Concentration Selection (units/mL)

100
▼

Calculate

ATTENTION: PLEASE FOLLOW THE PROMPTS BELOW

1. TURN OFF HEPARIN INFUSION and confirm the provider received panic value alert.
2. Repeat STAT aPTT Hourly until aPTT \leq 100 seconds
3. IF the second (the first hourly) aPTT remains $>$ 150seconds, verify provider received second panic value alert.
4. When aPTT is in target range (\leq 100 seconds) resume infusion at a rate that is 2mL/hour lower than the prior to turning off.
5. Repeat aPTT 6 hours after the new, lower infusion rate has started."

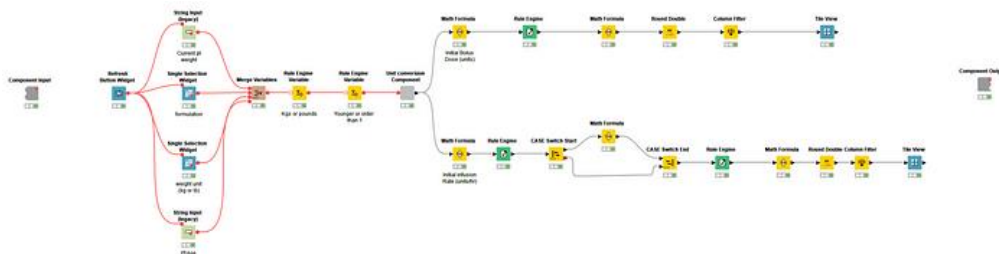
An example of the output text populated when the patient's measured aPTT is above 150 seconds utilizing our post-aPTT dose adjustment calculator.

2. Pediatric DVT/PE

A similar workflow is utilized for the pediatric DVT/PE initial bolus dose and initial infusion rate yet there is one discrepancy. The initial infusion dose is based on whether the child is less than or greater than one year of age; therefore, an additional *String Input* node is utilized for the patient's age.

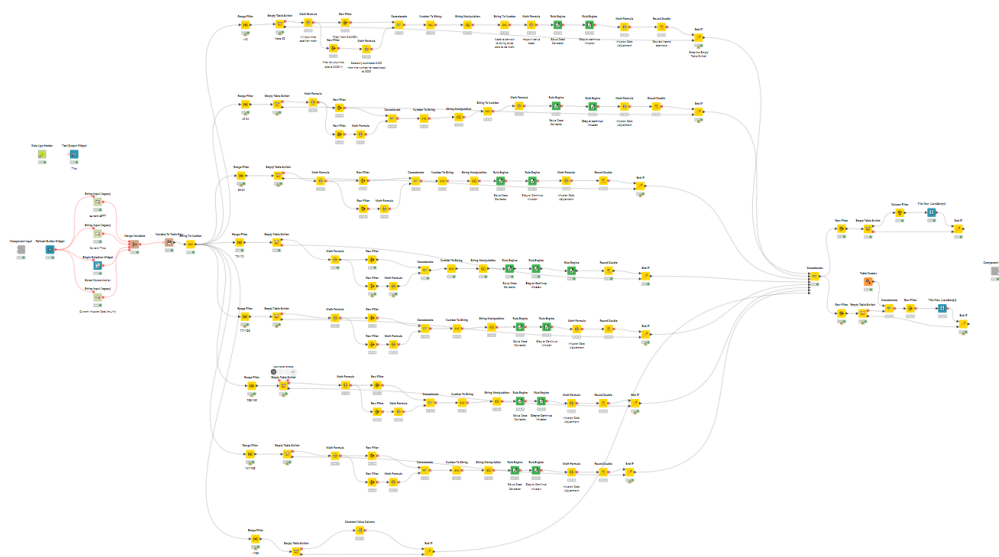
Data Apps for Healthcare Applications

Heparin Therapeutic Monitoring and Calculation with KNIME



This figure depicts the workflow used to calculate the heparin initial bolus dose and initial infusion rate in pediatrics. Noting the additional string input for patient age.

The aPTT ranges are slightly different from the ones seen in the adult general CV and DVT/PE calculators: <55, 55–69, 70–110, 111–125, 126–140, and >140. The therapeutic aPTT goal is broader at 70–110 seconds. In addition, the aPTT dose adjustments are based on rate change perchange.



This figure is an overview of the workflow to determine the post-aPTT dose adjustments for pediatric heparin dosing.

The calculator described above as well as other KNIME based workflows related to healthcare can be downloaded from our public [Digital Healthcare](#) from the KNIME Community Hub.

Watch the presentation on YouTube: [“2022 Fall - Data Science Elective Final Project - KNIME heparin calculator”](#).

Track Disease with KNIME on COVID-19 Dashboard

Author: [Ali Asghar Marvi](#) - KNIME

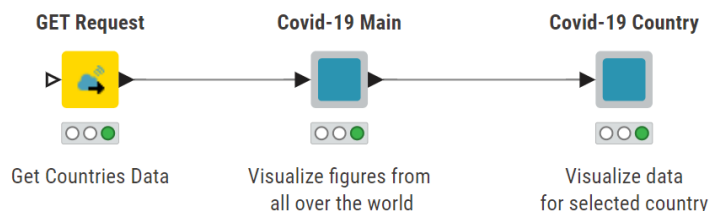
Workflow on KNIME Community Hub: [Covid-19 Dashboard Updated](#)

In early 2020, the world was hit by a deadly virus. It not only took millions of lives, but also crippled the global economy, bringing it to a standstill. Amid efforts to reduce the impact of the virus, Johns Hopkins University stepped in to track COVID-19 data from sources from across the world. It introduced a coronavirus tracker web service to query data, whether from the wider world or specific countries. This Swagger interface used to look through all web services definitions [can be found here](#).

There are two versions of these web services. The first version, or “V1,” has four REST endpoints, none of which take any input parameters. Anyone can query data for all countries for each date for the last two years, or simply get statistics on confirmed cases, recovered patients, or number of deaths reported so far. Version two, or “V2,” lets users query not only Johns Hopkins, but also the New York Times (NYT) and Conference of State Bank Supervisors (CSBS). However, NYT and CSBS return results for the USA only. Check out the [Github](#) documentation for a detailed description of the web services.

Let me walk you through how I access the V2 web service using the *GET Request* nodes in KNIME and the two COVID-19 components I've built to create dashboards to track this disease. This dashboard can be deployed as a so-called [data app](#), a browser-based application that can be used centrally or shared with others.

Tracking Disease in the COVID-19 Dashboard



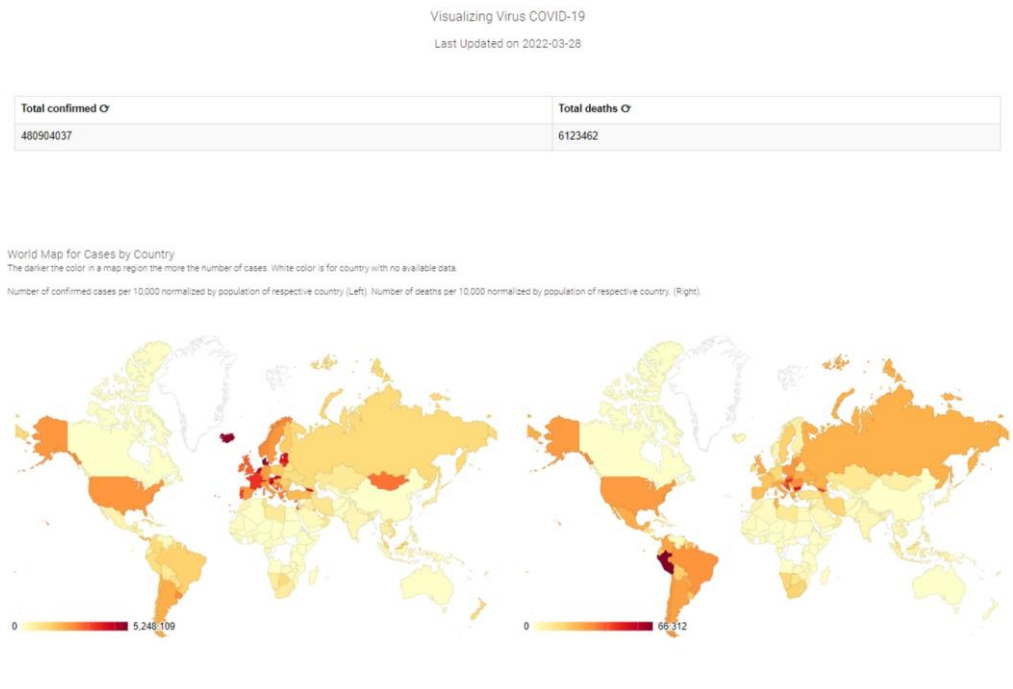
This figure depicts the abstract view of the dashboard workflow.

I make a data query in KNIME using multiple *GET Request* nodes. In the first node (figure above), I call the “V2 latest” webservice to get the latest numbers on reported cases and deaths, both for the world as a whole and each individual country. The

following component, “COVID-19 Main,” processes the output data and visualizes it. The second component, “COVID-19 Country,” calls another web service, “V2 locations,” to query country-specific data. The country is selected in the dropdown menu, and the corresponding data is visualized accordingly.

Overview of Confirmed COVID-19 Cases to Date

This dashboard has two parts. The “COVID-19 Main” component gives an overview of the total confirmed cases and total death cases to this day, and two choropleth maps show reported numbers normalized with respect to the population of the country up to the last 24 hours. One map reflects the confirmed cases, and another the deaths so far (figure below).



Output of “COVID-19 Main” dashboard view.

Visualize Country-Specific COVID-19 Data

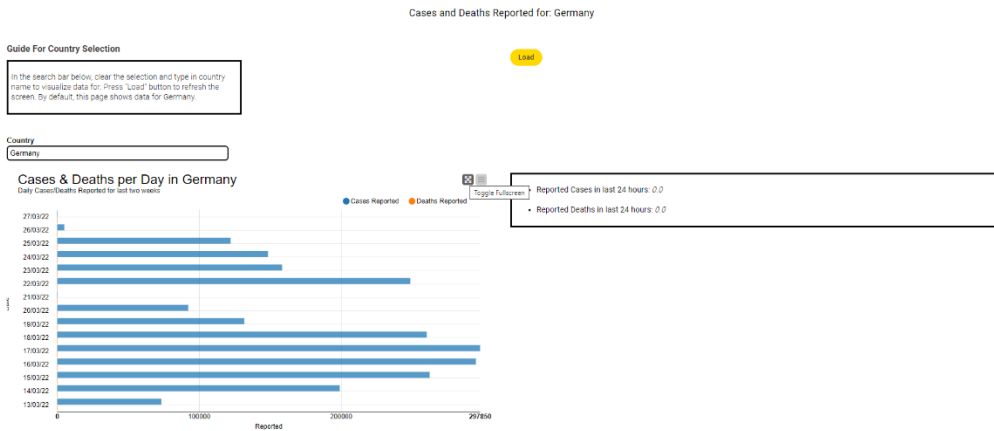
The “COVID-19 Country” component returns country-specific data for the past two years. This dashboard has four important segments:

- numbers reported each day over the last two weeks.
- numbers reported in the last 24 hours

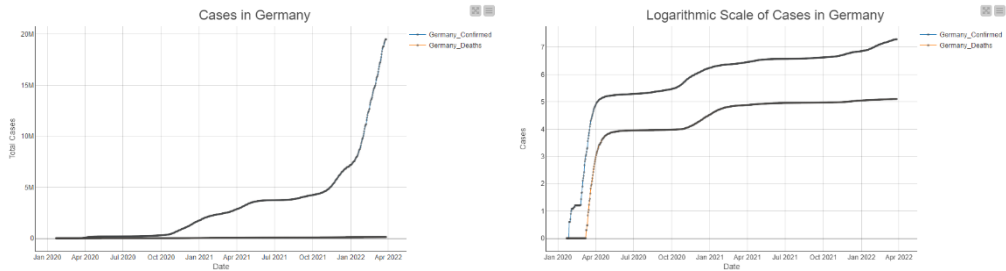
Data Apps for Healthcare Applications

Track Disease with KNIME on COVID-19 Dashboard

- interactive line plot of cases in the selected country (left)
- logarithmic scaled line plot of cases reported (right)



This dashboard is visualized through a browser-based data app, but it can be also visualized locally on your machine as a local "Interactive View".



This dashboard is inspired by [COVID-19 Live Visualization using Guided Analytics](#) and [Visualize COVID-19 in Italy](#).

Analyzing Digital Healthcare Data

KNIME Analytics Platform supports data access from a wide variety of sources, both local and remote. This includes simple tabular files, databases, and even data warehouses. The visual workflows document how data is loaded but also provide an abstract understanding of how the data is blended – independent of size or type. The possibilities (not only in digital healthcare use cases) to load and process data are endless, especially concerning patient-specific data sources such as FHIR services, or open databases like Federal Adverse Event Reporting System (FAERS) and Vaccine Adverse Event Reporting System (VAERS). Once data is loaded, KNIME's full potential can be exploited in terms of exploring the data, statistical analysis, creating predictive models, and deploying them as data apps or web services.

This chapter includes the articles:

- **Interact with Epic on FHIR to visualize Patient Data**, p. 36
 - Franziska Rau, *KNIME &*
 - Ali Asghar Marvi, *KNIME*
- **How to use FAERS for Adverse Reaction Management**, p. 42
 - Lama Basalelah, *Virginia Commonwealth University*
 - Suad Alshammari, *Virginia Commonwealth University*
 - Joshua Morriss, *Virginia Commonwealth University*
 - Dayanjan Wijesinghe, *Virginia Commonwealth University &*
 - Franziska Rau, *KNIME*
- **Using KNIME to undertake Pharmacovigilance of Special Patient Populations**, p. 46
 - Noah Frazier, *Virginia Commonwealth University &*
 - Dayanjan Wijesinghe, *Virginia Commonwealth University*

Interact with Epic on FHIR to visualize Patient Data

Author: [Franziska Rau](#) & [Ali Asghar Marvi](#), KNIME

Workflows on the KNIME Community Hub: [Epic on FHIR Demo Dashboard](#)

Healthcare data is complex. It is stored in multiple places, in multiple computer systems, and comes from multiple departments. This makes it difficult to use, measure, and share.

The Fast Healthcare Interoperability Resource (FHIR) was defined to improve data interoperability. It is one of the most popular protocols for joining disparate systems and it facilitates exchanging and sharing information on clinical health data and other electronic health records. It allows interoperability among healthcare systems to provide information to organizations and individuals on a wide variety of smartphones or computers. It also enables third-party vendors to create their own custom applications based on FHIR standards.

In this article, we'd like to show how you can build a browser-based application with KNIME that accesses healthcare data via the FHIR-based API that is provided by Epic on FHIR.

Application-based Interoperability for Seamless Data Exchange

Many electronic health record systems utilize the FHIR standard and provide free resources for developers to create healthcare apps for patients and healthcare organizations.

Note: FHIR utilizes an application programming interface (API) to exchange electronic health records (EHR) and electronic medical records (EMR). The protocol uses common standards such as JSON, XML, or RDF on REST-based protocols.

[Epic on FHIR](#) or [Cerner](#) are examples of such free resources for developers. They provide not only FHIR-based resources, but also a complete set of additional features like dedicated servers, cloud services, sandbox data to play with, and much more.

The Epic-on-FHIR Sandbox Environment

In this article, we will talk about how we accessed example data using the FHIR-based APIs provided by Epic on FHIR and consolidated and visualized it in a dashboard.

The [Sandbox Test Data](#) by Epic consists of a set of patients, each with a unique FHIR ID and corresponding resources of their medical history. In the screenshot below you can see there is a test patient called Derrick Lin with a FHIR ID and portal login credentials for the sample patient application.

Application credentials are for testing by developers, however we want to talk more about the applicable resources against the FHIR ID, and demonstrate how we called some of them in KNIME and visualized it all in a consolidated dashboard.

Patient	FHIR Identifier	Applicable Resources
Camila Lopez	FHIR: erXuFYUfucBZaryVksYecMg3 External: Z6129 MRN: 203713 MyChart Login Username: fhircamila MyChart Login Password: epicepic1	DiagnosticReport Medication MedicationOrder MedicationRequest MedicationStatement Observation (Labs) Patient Procedure
Derrick Lin	FHIR: eq081-VQc9P8dAUUqCWzHfw3 External: Z6127 MRN: 203711 MyChart Login Username: fhirderrick MyChart Login Password: epicepic1	CarePlan Condition Goal Medication MedicationOrder MedicationRequest MedicationStatement Observation (Smoking History) Patient

Sample of FHIR patients from the sandbox environment provided by EPIC.

How to Call Epic-on-FHIR Web Services in KNIME

The first and most important step is to create an account on Epic on FHIR. Once that's done, head to the "API Specifications" tab on the menu bar to see the list of available APIs (figure below).

For our example, we will only call four web services:

- Patient.Read (R4) - basic patient information like name, age, address and more
- Condition.Search (R4) - all past conditions of a patient
- MedicationRequest.Search (Orders) (R4) - medication prescription based on conditions
- Goal.Search (Patient) (R4) - medical prescription of various therapies

Each service is based on the R4 version of HL7 for data interoperability. It takes the patient's FHIR ID as input, and the relevant data is returned in XML format. It is subsequently converted into JSON format to extract the necessary fields, and finally, visualized in KNIME.

Analyzing Digital Healthcare Data

Interact with Epic on FHIR to visualize Patient Data

The screenshot shows the Epic FHIR API Specifications page. On the left, there is a search bar and a list of 360 APIs. The right pane displays details for the **AllergyIntolerance.Read (R4)** API, including its general information, description, request, and response.

General Information

- Http Method: GET
- Url Template: api/FHIR/R4/AllergyIntolerance/{ID}

Description

Returns data about an allergy or intolerance to a specific substance associated with one patient. You can also search by ID and by patient, allowing it to return a list of allergies

This API may behave differently when used in a patient-facing context. See the [Patient-Facing Apps Using FHIR](#) document for more information.

Request:

```
/Interconnect/api/FHIR/R4/AllergyIntolerance/ee2xw9V18-QmUQuidH8K1u3
```

Response:

Name	Description	Is Optional	Is Array
ID (String)	The AllergyIntolerance FHIR ID.	false	false

```
<AllergyIntolerance xmlns="http://hl7.org/fhir">
  <id value="ee2xw9V18-QmUQuidH8K1u3" />
  <clinicalStatus>
    <coding>
      <system value="http://terminology.hl7.org/CodeSystem/allergyintolerance-clinical" />
      <version value="4.0.0" />
      <code value="active" />
      <display value="Active" />
    </coding>
  </clinicalStatus>
  <verificationStatus>
    <coding>
      <system value="http://terminology.hl7.org/CodeSystem/allergyintolerance-verification" />
      <version value="4.0.0" />
      <code value="confirmed" />
      <display value="Confirmed" />
    </coding>
  </verificationStatus>
  <category value="medication" />
  <code>
    <coding>
      <system value="urn:oid:2.16.840.1.113883.6.88" />
      <code value="9835" />
      <display value="SHALLPOX VACCINE" />
    </coding>
    <coding>
      <system value="urn:oid:2.16.840.1.113883.4.9" />
      <code value="49V998895K" />
    </coding>
  </code>
  <text value="SHALLPOX VACCINE" />
  <code>
    <coding>
      <system value="https://hostname/instance/api/FHIR/R4/Patient/e65bT0QqbCKHIq2o98DA3" />
      <display value="Lurch, William" />
    </coding>
  </patient>
  <recordedDate value="2019-04-19" />
</AllergyIntolerance>
```

Snippet showing the huge list of web services available (left) in EPIC on FHIR

Web Services for Patient Conditions, Prescriptions, and Health Goals

Web services or APIs are called into KNIME and parsed to extract the information we want to visualize. In our example, we used the FHIR ID of the test patient, Derrick Lin. This patient is fictitious and is from the Sandbox environment provided by Epic.

In order to call each web service, Epic requires a bearer token to be sent as header. You'll see the bearer token in the Raw Request (below) of any selected API from the list (above). The workflow requires that you add this token in the configuration window, and it is propagated as a flow variable for each web service call.

Analyzing Digital Healthcare Data
Interact with Epic on FHIR to visualize Patient Data

Http Method
GET

URL Template
`https://fhir.epic.com/interconnect-fhir-oauth/api/FHIR/R4/Patient/{ID}`

Request Parameters
ID: `e63wRTbPfr1p8UW81d8Seiw3`

Raw Request
GET `https://fhir.epic.com/interconnect-fhir-oauth/api/FHIR/R4/Patient/e63wRTbPfr1p8UW81d8Seiw3`
Authorization: Bearer `eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWQiOiJlc46b2lkcmZoaXNlLCJjYjG1bnRfawQio`

Try It Out!

Bearer token from trying out a patient's web service.

Let's now have a closer look at these web services.

Patient.Read (R4) for Basic Patient Information.

This web service is called in KNIME using the *GET Request* node. The *xml*/returned is converted to JSON format, and necessary fields are picked using the *JSON Path* node. The extracted fields include "Official Name," "Date of Birth," "Sex," etc. This is the demographic and administrative information of the person receiving healthcare from the organization.

This is the REST endpoint for the patient. The part highlighted blue is the FHIR ID of the patient.

```
https://fhir.epic.com/interconnect-fhir-oauth/api/FHIR/R4/Patient/eq081-VQEGP8drUUqCWzHfw3
```

Condition.Search (R4) for a Patient's Previous Conditions

Similarly to calling a patient's rest-based endpoint, the "Condition" web service searches for the condition and diagnosis of a given patient, along with their clinical status and reason for visit – in this case for "Derrick." The parsed data is visualized inside the view component. This web service takes in a patient's FHIR ID as a search parameter.

This is the REST endpoint to search for a condition of a given patient.

```
https://fhir.epic.com/interconnect-fhir-oauth/api/FHIR/R4/Condition?patient=eq081-VQEGP8drUUqCWzHfw3
```

MedicationRequest.Search (Orders) (R4) for Medication Prescriptions

Based on the diagnosis, a patient is prescribed medication or any relevant therapy along with dosage by the practitioner. In this case, this web service returns the medication request by the patient. This could be used by relevant pharmacies or medical institutions to keep track of a patient's consumption. The rest endpoint (in screenshot below) takes in the patient's FHIR ID as a search parameter in the form of a query parameter.

The REST endpoint to search for a prescribed medication of a given patient.

```
https://fhir.epic.com/interconnect-fhir-oauth/api/FHIR/R4/MedicationRequest?patient=eq081-VQEGP8drUUqCWzHfw3
```

Goal.Search (Patient) (R4) for the Targeted Health Outcome

Every prescribed medication has an end goal associated with a patient: Obviously, an improvement of their existing conditions. The "Goal" rest service is called to identify desired health outcomes associated with Derrick. This rest point also takes in the patient's FHIR ID as a search input (see below).

The REST endpoint to search desired health goal of a given patient.

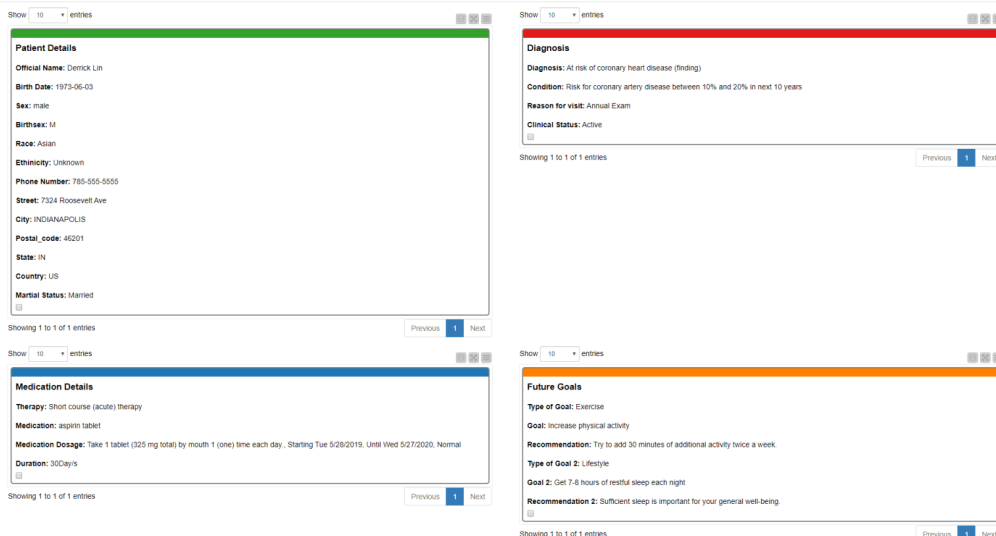
```
https://fhir.epic.com/interconnect-fhir-oauth/api/FHIR/R4/Goal?patient=eq081-VQEGP8drUUqCWzHfw3
```

Visualizing Healthcare Data in a Dashboard

After consuming all the rest points and parsing the necessary information, we visualized it using Tile View for each category. All tiles are put together inside a component and organized to view it in one consolidated dashboard. To learn more about creating a dashboard in KNIME, [this blog](#) will help you understand its fundamentals.

Analyzing Digital Healthcare Data

Interact with Epic on FHIR to visualize Patient Data



Consolidated visualization of data from each EPIC on FHIR service.

An EPIC-on-FHIR Patient Dashboard

In summary we can say that we have successfully accessed several EPIC on FHIR services in KNIME to visualize a sample patient from a sandbox environment.

However, it is important to note that for production-based environments, any health institution using EPIC on FHIR needs to have a middleware to cater for authentication issues – such as getting access tokens in this case. The visualization in KNIME will be the final step.

Try out our example workflow for this dashboard and download it from the KNIME Hub: [EPIC on FHIR \(Patient Dashboard\)](#).

How to use FAERS for Adverse Reaction Management

Authors: [Dayanjan Wijesinghe](#), [Lama Basaleleh](#), [Suad Alshammari](#), & [Joshua Morriss](#) - Virginia Commonwealth University, [Franziska Rau](#), KNIME

Workflows on the KNIME Community Hub: [FAERS Data Download](#) and [FAERS Data Processing Loperamide](#)

Hypothesis Testing on FAERS Data and Using it for Adverse Reaction Management

Every quarter the FDA (Food and Drug Administration) provides reports about adverse events, medication errors, and product quality complaints. The system used for that is the Federal Adverse Event Reporting System (FAERS) which is a database designed to support the FDA's pharmacovigilance program for adverse events in response to drug and therapeutic biologic products and medication errors. Adverse events are associated with the use of medical products leading to unintended symptoms or diseases, often with harmful consequences.

We wanted to demonstrate a proof of concept application for accessing and analyzing FAERS data to test various hypotheses. Here we demonstrate the use of KNIME and FAERS to investigate the effects the medications Irinotecan and Loperamide have on Intestinal toxicity.

Irinotecan Medication for Cancer Treatment

Irinotecan has been used for the treatment of a number of solid tumors, including lung, colorectal, and pancreatic tumors. Its primary mode of action is via the inhibition of the topoisomerase I enzyme, which is responsible for DNA replication and transcription that causes cancer cells to survive and proliferate⁷.

One of the serious dose-related toxicities is intestinal toxicity, in which a patient suffers from life-threatening conditions like diarrhea, depletion in fluid and electrolytes, and hospitalization.

⁷ de Man FM, Goey AKL, van Schaik RHN, Mathijssen RHJ, Bins S. Individualization of Irinotecan Treatment: A Review of Pharmacokinetics, Pharmacodynamics, and Pharmacogenetics. *Clin Pharmacokinet*. 2018 Oct;57(10):1229-1254. doi: 10.1007/s40262-018-0644-7. PMID: 29520731; PMCID: PMC6132501.

Intestinal toxicity has two phases:

- Early-onset diarrhea, which usually happens within 24 hours. This usually happens due to the cholinergic properties and is usually accompanied by other cholinergic symptoms like abdominal cramps.
- Late-onset diarrhea occurs 24 hours after administration. The luminal irinotecan metabolites induce mucosal damage-causing malabsorption and imbalance with water and electrolytes.

Eventually, the gut-bacterial β -glucuronidase converts the irinotecan metabolite SN38G into an active form of SN38 that causes luminal damage and diarrhea⁸.

Thus, a question we can ask is: Are there any medications, which, when administered with Irinotecan, have the potential to negate the onset of intestinal toxicity?

To answer this question, we tested the hypothesis that Irinotecan-driven intestinal toxicity is negated by the co-occurrence of Loperamide, a treatment used to treat diarrhea, which is discussed in the paper⁹.

Our main hypothesis for testing is:

1. The number of Irinotecan-driven Intestinal toxicity cases is reduced with co-occurrence Loperamide

Which leads us to two sub-hypotheses:

1. The count of Loperamide use should be lower in those reporting intestinal toxicity and Irinotecan
2. The count of Loperamide use should be lower in those reporting Intestinal toxicity-related events and Irinotecan.

Hypothesis Testing on Adverse Events

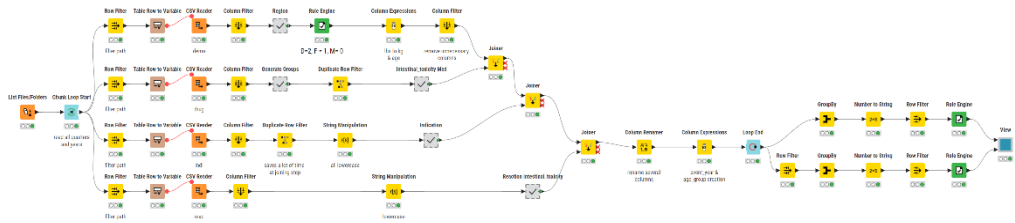
First of all, we downloaded the data from the [FDA website](#). The data from every quarter needs to be downloaded separately and contains multiple files including demographic information, drug and reaction information from case reports and patient outcome information.

⁸ Stein A, Voigt W, Jordan K. Chemotherapy-induced diarrhea: pathophysiology, frequency and guideline-based management. *Ther Adv Med Oncol*. 2010 Jan;2(1):51-63. doi: 10.1177/1758834009355164. PMID: 21789126; PMCID: PMC3126005.

⁹ Alimonti A, Gelibter A, Pavese I, Satta F, Cognetti F, Ferretti G, Rasio D, Vecchione A, Di Palma M. New approaches to prevent intestinal toxicity of irinotecan-based regimens. *Cancer Treat Rev*. 2004 Oct;30(6):555-62. doi: 10.1016/j.ctrv.2004.05.002. PMID: 15325035.

Analyzing Digital Healthcare Data

How to use FAERS for Adverse Reaction Management



View of the overall workflow.

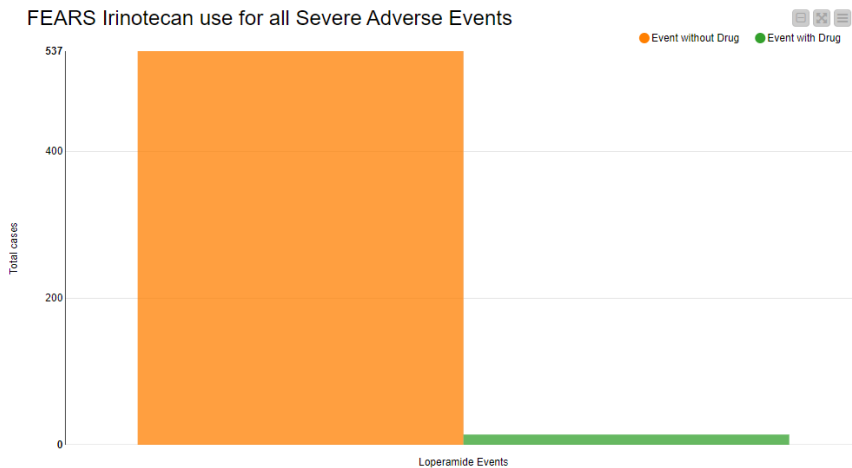
After reading in all the different CSV files, we grouped the patients by medication to get a general overview.

Group 1 includes all patients that took Irinotecan, Group 2 Irinotecan and Loperamide and Group 3 only Loperamide as medication.

We took data from 2015 through to Q3/2021. It indicated that simultaneous intake prescription of Irinotecan and Loperamide is not as common as taking both medicines by themselves.

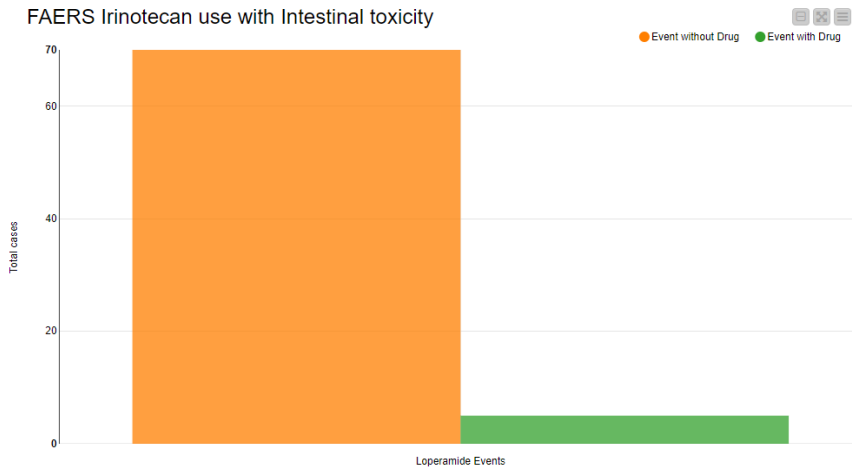
Afterwards we counted all the severe adverse events for both groups. As you can see in the following figure, the patients who only took Irinotecan showed significantly more adverse events than those who took Irinotecan and Loperamide in combination.

The analysis also shows us that more patients suffered from intestinal toxicity when using only Irinotecan as medication in comparison with patients who also took Loperamide in combination, seen in figure below.



Severe adverse events for Irinotecan only and in combination with Loperamide.

Analyzing Digital Healthcare Data
How to use FAERS for Adverse Reaction Management



Cases of hypertension using Irinotecan without and with Loperamide.

Download and Query of FAERS Data Easily with KNIME

In summary, the workflow allows us to easily query the downloaded FAERS data for specific hypotheses testing. Download our [publicly available workflows](#) to use as your own starting point for processing FAERS data.

Using KNIME to undertake Pharmacovigilance of Special Patient Populations

Authors: [Dayanjan Wijesinghe](#) & [Noah Frazier](#) - Virginia Commonwealth University

Workflow on the KNIME Community Hub: [Using VAERS for pharmacovigilance of special populations](#)

Note. Please use this story only as a guide to create your own workflows with [KNIME](#). The results of this brief analysis have not been validated by independent review, and is only meant to be used as a demonstration of the capabilities of [KNIME](#) for pharmacovigilance towards vaccines in special populations.

Adverse events are an undesirable effect of a drug or other type of treatment. They can range in severity from mild to life threatening. The U.S. Food and Drug Administration (FDA) and drug manufacturers regularly monitor a drug's adverse events during clinical trials and post FDA approval. Post-marketing surveillance is important because there is a greater diversity in patients taking the medication compared to the clinical trial population. An increase in frequency of adverse events reported can result in a drug label change or even removal from the market.

The [Vaccine Adverse Event Reporting System \(VAERS\)](#) is a database of reported adverse events following immunization from U.S. licensed vaccines. VAERS serves the purpose of an early warning system for the U.S. to ensure vaccines are a safe tool for immunization. [Us and others have previously demonstrated in multiple instances the data mining capabilities of KNIME with respect to healthcare](#). Thus, we wanted to test the feasibility of using KNIME to monitor adverse events from special populations in response to vaccines for use as a post market surveillance tool.

Special populations are often excluded from clinical trials due to the complexity they may present when interpreting clinical trial results. With rigorous pharmacovigilance tools, we can identify trends and prevent potential harm to patients with unique underlying health conditions. In our proof of concept, we wanted to monitor what adverse events have been reported to VAERS from patients who have been diagnosed with multiple sclerosis and received the COVID-19 vaccine.

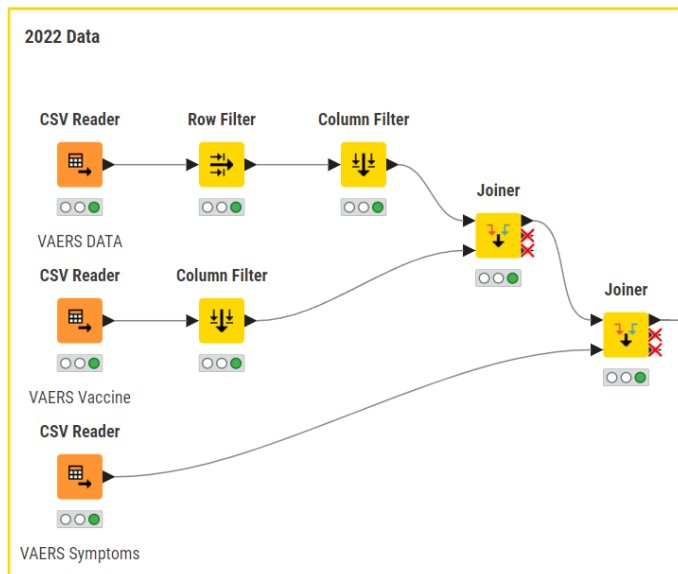
Multiple Sclerosis and the COVID Vaccine

Multiple sclerosis is a neurological disease that eats away at the protective covering of nerves. MS ultimately disrupts the communication pathway in the nervous system: potentially resulting in a disabled individual.

The COVID vaccines were designed as a tool to combat the COVID-19 pandemic our society was faced with. Given the situation of the crisis, the COVID vaccines were developed and approved in the US in an accelerated fashion. It is important for regulators, healthcare providers, and patients to be able to characterize the safety of the COVID vaccines for patients with multiple sclerosis.

KNIME Workflow

To start out, we downloaded the adverse events that were reported for the year 2022 from VAERS. The files were too large for an Excel spreadsheet to be used, so the data was saved as a CSV format. VAERS segments their data into 3 different files. One file pertains to patients' demographics, another file provides information on the vaccine received, and the last file contains data about the adverse events reported. The tables were joined and the rows were aligned by the unique VAERS ID each report receives.



VAERS data tables were combined by aligning their unique VAERS ID.

The data was cleaned in a series of steps of removing duplicate rows, rows with missing data, and columns that did not pertain to this analysis.

Analyzing Digital Healthcare Data
Using KNIME to undertake Pharmacovigilance of Special Patient Populations

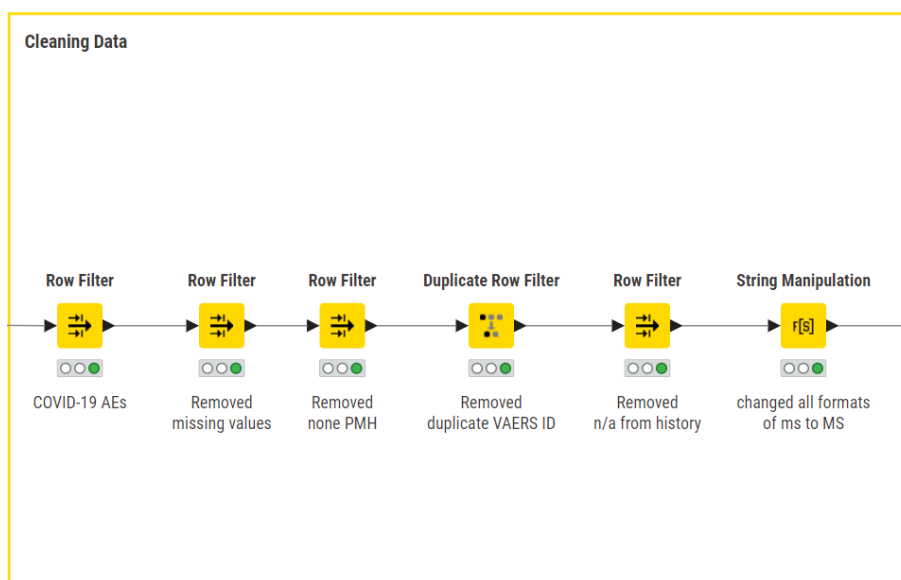


Illustration of steps needed to clean VAERS data.

To identify our special population of patients with multiple sclerosis, we had to standardize the spelling of multiple sclerosis to “MS”. Since VAERS reported incidents are inputted from the public, there were different spellings of multiple sclerosis i.e., “MS”, “ms”, “multiple sclerosis”, or “Multiple Sclerosis”. After standardizing the spelling

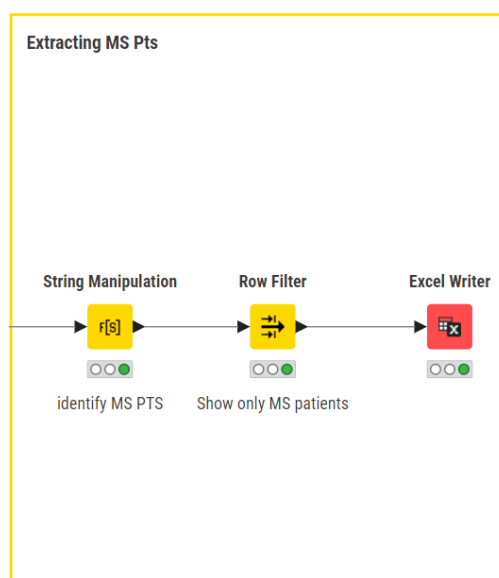


Illustration of how using the string manipulation node, we were able to standardize the spelling of multiple sclerosis to MS. The string manipulation node was also used to identify our special population data.

Analyzing Digital Healthcare Data *Using KNIME to undertake Pharmacovigilance of Special Patient Populations*

of multiple sclerosis and identifying which rows contained MS, we filtered out all unnecessary rows.

There are many ways to analyze the data of adverse events reported. As a starting point, we wanted to quantify the adverse events reported for patients with multiple sclerosis that received the COVID vaccine. The data was configured in a way so it would be compatible with the *Bag of Words Creator* node. This node was chosen because it breaks down the data in an easy to quantify format. The data output of this node was exported to an Excel sheet where a pivot table was created.

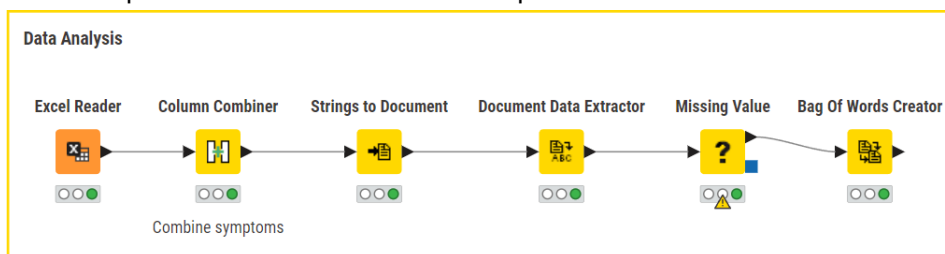
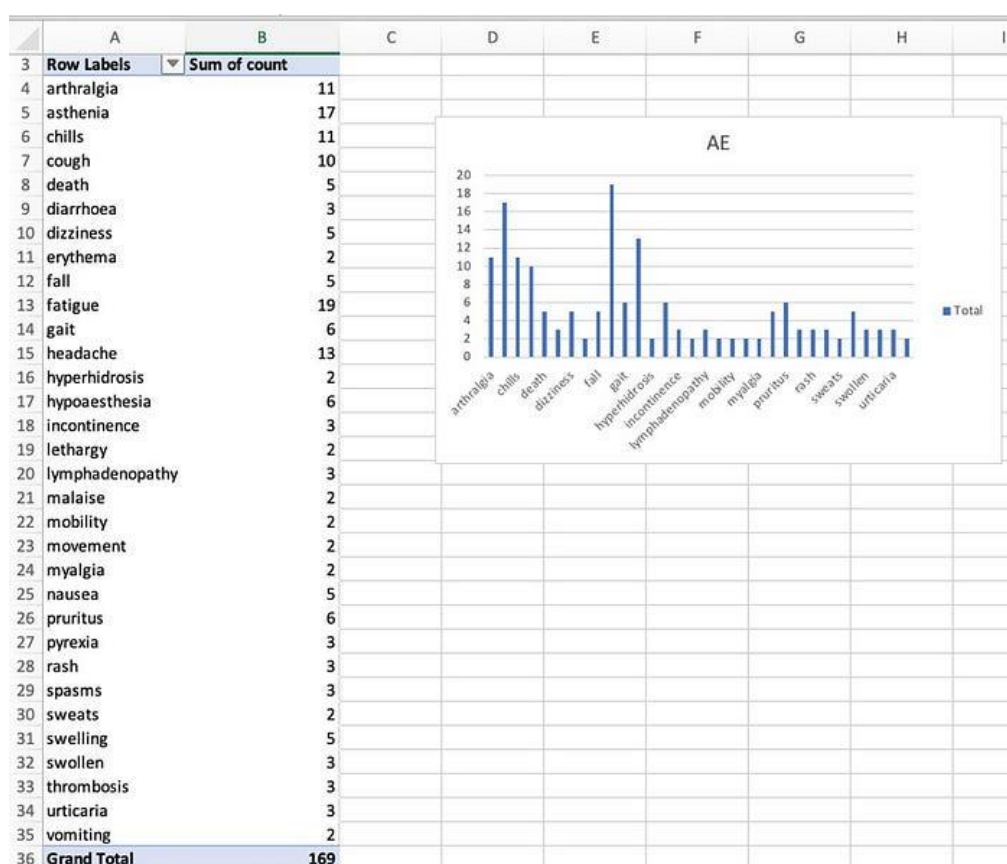


Illustration of steps needed to configure data in format compatible with the Bag of Words Creator node.



Results from the pivot table.

Analyzing Digital Healthcare Data
Using KNIME to undertake Pharmacovigilance of Special Patient Populations

Overall, this exercise demonstrates the ease by which KNIME can be utilized to create pharmacovigilance tools for special populations.

A video demonstration of Dr. Noah Frazier demonstrating how he is using KNIME to undertake the above workflow on YouTube: "[Using KNIME to undertake Pharmacovigilance of Special Patient Populations](#)".

Deep Learning for Digital Healthcare

While KNIME provides a plethora of statistical and machine learning methods out-of-the-box, it also allows users to effortlessly benchmark methods and integrate cutting-edge technologies like deep learning into their projects. With its two deep learning extensions, based on Keras and Tensorflow, users cannot only apply or finetune pre-trained models but also build and train their custom networks without coding.

This chapter shows how codeless deep learning can benefit digital healthcare applications like predicting blood glucose levels or discriminating normal and abnormal ECG for detecting arrhythmia.

This chapter includes the articles:

- **Predict Blood Glucose with an LSTM on CGM Data**, p. 52
 - Franziska Rau, *KNIME*
- **Learn how to classify Electrocardiogram Signals with Deep Learning in KNIME**, p. 59
 - Ali Asghar Marvi, *KNIME*
- **How to perform Electrocardiogram Categorization and Detect Arrhythmia**, p. 67
 - Ali Asghar Marvi, *KNIME*

Predict Blood Glucose with an LSTM on CGM Data

Author: [Franziska Rau](#), KNIME

Workflow on the KNIME Community Hub: [Glucose Level Prediction](#)

Processing Continuous Glucose Monitoring (CGM) Data for Digital Healthcare on KNIME Hub

The number of people suffering from diabetes is increasing drastically worldwide. Devices called **Continuous Glucose Monitors (CGMs)** exist to help make the everyday lives of diabetics easier. These tiny sensor wires are inserted under the skin, and then record blood sugar (glucose) levels at certain time intervals, transmitting the data to an app or software. This way, a user can see their current and past blood glucose levels. But wouldn't it be great to know your *future* levels as well?

To that end, I trained an LSTM network with existing CGM data to predict future blood glucose levels. This makes it possible to recognize early on whether your sugar levels could rise or fall drastically, enabling rapid intervention. It's a great way to make everyday life easier for diabetics, and I'll show you how to do it in KNIME. You can already see the complete workflow in the figure below.

Glucose Level Prediction

This workflow reads in the diabetes subset from the Dubosson data set, which contains glucose measurements, as well as several sensor measurements like heartrate, breathing rate, activity and acceleration.

The exploration Component generates an overview of all the different measurements and lets the user filter the data by date and patient.

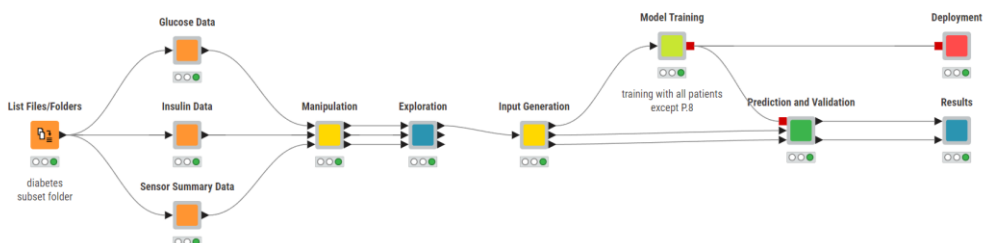
Afterwards an LSTM network is trained using 15 minutes of blood glucose measurements for training. The goal is to predict the next 15 minutes of blood glucose levels for each patient.

The data is split into training and test set by taking 80% of the data for each patient from the top as training set and the remaining 20% as test set. "Diabetes_Patient_008" is not included in the training/testing process. This patient will be used for validation purposes.

At the end, the model performance is displayed in an interactive view.

The data is available for download from:

- https://zenodo.org/records/5651217/files/diabetes_subset_pictures-glucose-food-insulin.zip?download=1
- https://zenodo.org/records/5651217/files/healthy_subset_sensor_data.zip?download=1



The whole prediction workflow, containing data reading, feature exploration, input generation, model training, and testing and deployment.

Continuous Glucose Monitoring

There are two types of diabetes: Type 1, in which the body does not produce enough or any insulin, and Type 2, in which the body doesn't use insulin properly¹⁰. Type 2 is the most common form of diabetes, and some people can manage their blood sugar levels with a healthy diet and exercise. Others may need insulin to help control their blood sugar levels. For them, keeping track of their blood sugar is a crucial everyday task.

In this post, we will look at the D1NAMO dataset, which contains CGM blood glucose measurements and insulin dose data from Type 1 diabetes patients. The cool thing is that it contains data not only from CGMs, but also motion sensors. The participants were each monitored using the Zephyr BioHarness 3 wearable chest belt device, which recorded information on their hearts, respiration rates, and body acceleration.

Currently, software/apps which use CGMs to display one's blood glucose levels can only show the past and current values. But wouldn't it be helpful to know the future values as well? With this information, a diabetic could know if their blood sugar level will soon rise or sink. In my workflow, I used an LSTM network to make these kinds of predictions.

Visualizing CGM and Sensor Data

To explore the different features, I created an interactive view via a component which shows the different measurements, as seen in following figure. It shows measurements from "Diabetes_Patient_1" on October 1st, but you can also view other patients and dates. This is possible by combining several [Line Plots](#) that operate on the same data table and additional [Value Selection Widgets](#), which allow me to filter the data directly in the view. I also used a new widget, the [Refresh Button](#). With this, you no longer have to close your view to apply new settings.

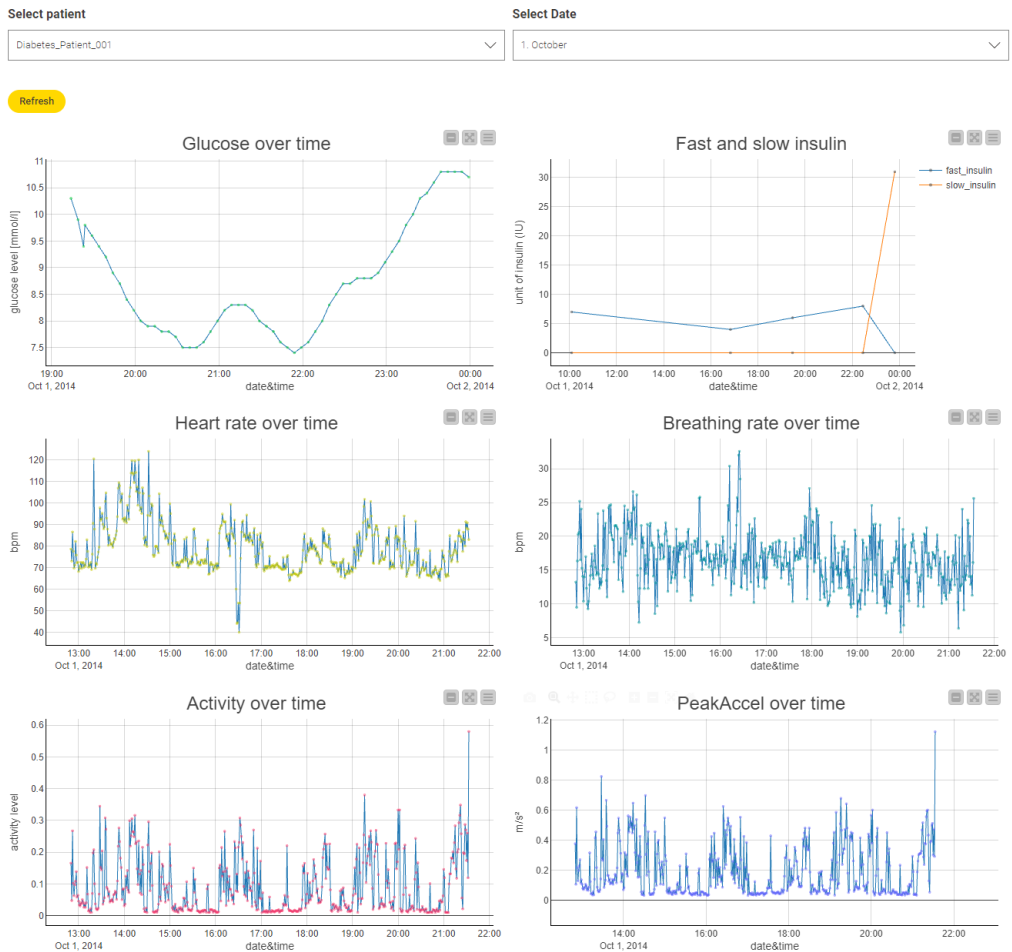
I included the features "heart rate," "breathing rate," "activity," and "peak acceleration over time" in this view. "Activity" and "peak acceleration" refer to the mean and maximum acceleration magnitude measured by the sensor. Insulin values are also included, showing how much fast or slow insulin was injected at which time.

But the dataset includes more sensor data as well. If you are interested in exploring the whole set or learning more, read "The open D1NAMO dataset: A multi-modal dataset for research on non-invasive type 1 diabetes management" from

¹⁰ *Diabetes Symptoms, Causes, & Treatment. (n.d.). American Diabetes Association from <https://www.diabetes.org/diabetes>*

Deep Learning for Digital Healthcare
Predict Blood Glucose with an LSTM on CGM Data

Dubosson¹¹. To keep things simple, I continued with only the blood glucose values for my prediction, because usually diabetic patients don't wear any monitoring devices besides a CGM. But if you like challenges, you could try to use all the features for your own prediction.



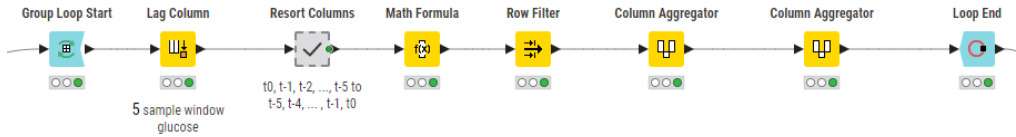
Interactive view of the CGM and sensor data. Within this view, you can select a patient and a date to look at.

Blood Glucose Prediction using Many-to-Many LSTM

Before we look at the model, we have to generate the appropriate input data.

¹¹ Dubosson, F. (2018). The open D1NAMO dataset: A multi-modal dataset for research on non-invasive type 1 diabetes management. *Informatics in Medicine Unlocked*, Volume13(-), Pages92-100. <https://www.sciencedirect.com/science/article/pii/S2352914818301059>

A helpful post by [Kathrin Melcher](#) shows an example of [Multivariate Time Series Analysis with LSTMs](#) using a Many-To-One network architecture. She explains the input generation and the LSTM model in more detail. I used a "Many-To-Many" model architecture, which needs the input and target values as sequences of vectors. Figure below shows the "Input generation" component content.



Input generation for a "Many-to-Many" LSTM architecture.

At each iteration, the data of one patient is processed. The [Lag Column](#) node in the loop body creates the sequence of $n=5$ past values of the current column by setting a lag value of 5. The times series produced by the *Lag Column* node follows this order:

At each iteration, the data of one patient is processed. The *Lag Column* node in the loop body creates the sequence of $n=5$ past values of the current column by setting a lag value of 5. The times series produced by the *Lag Column* node follows this order:

$$x_{t_0} \ x_{t_0-t_1} \ x_{t_0-t_2} \ x_{t_0-t_3} \ x_{t_0-t_4} \ x_{t_0-t_5}$$

The network needs a sequence in time increasing order, such as:

$$x_{t_0-t_5} \ x_{t_0-t_4} \ x_{t_0-t_3} \ x_{t_0-t_2} \ x_{t_0-t_1} \ x_{t_0}$$

The *Resort Columns* metanode resorts the sequence appropriately. The output of the *Loop End* node has the sorted sequences of feature vectors and the corresponding target vectors.

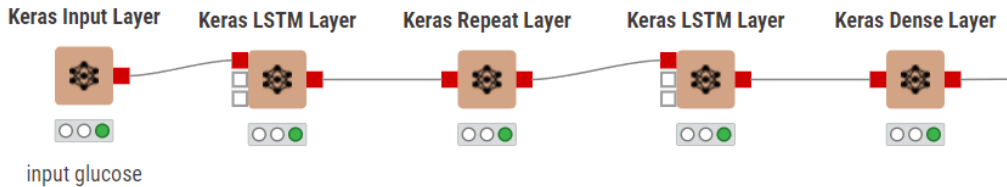
In my example, the input vectors contain samples with three time steps each, while the output has the next three consecutive timesteps. This corresponds to a 15-minute input window and a 15-minute prediction window.

Afterward, the data is split into training and test sets by taking 80% of the data for each patient from the top as training set and the remaining 20% as test set. Keep in mind that since this is time series data, you don't want to take random samples. Additionally, I did not include "Diabetes_Patient_008" in the training/testing process. This patient will be used for validation purposes.

The Long Short-Term Memory (LSTM) network is a type of recurrent neural network, often used in deep learning because even very large architectures can feasibly be trained with it. It is commonly used for time series prediction.

Instead of neurons, LSTM networks have memory blocks that are connected through layers. A block has components that make it smarter than a classical neuron and give it a memory for recent sequences. I won't go into detail, but if you are interested, read this great blog post about [LSTMs](#).

The network I used consists of five layers, as seen in figure below:



Different Keras Layers that make up the final model.

The five layers in detail:

1. An input layer to define the input shape implemented via a [Keras Input Layer](#) node. The input shape is a tuple, represented as n, m, where n is the length of the sequence and m is the size of the feature vector at each time step. In this example, the input shape is 3,1 (three time steps of 5 minutes and one feature (glucose level)).
2. An LSTM layer implemented via a [Keras LSTM Layer](#) node, using 100 units (length of the vector in hidden state) and "ReLu" as its activation function.
3. A [Repeat Layer](#) that repeats the input three times. The Repeat Layer adds extra dimensions to the dataset.
4. A second LSTM layer using 100 units and "ReLu" activation function. This time the "Return sequences" option is enabled (this returns the hidden state output for each input time step).
5. A [Dense Layer](#) that connects each unit of the layer input with each output unit of this layer. This generates an output of the shape 3,1 (like the input shape)

Using a [Keras Network Learner](#) node, the created model can be trained. I trained the model for 100 epochs, and then used the [Keras Network Executor](#) node on the test and validation set.

To view the results, I again created an interactive view, which makes it possible to select the different patients and plot the corresponding result. Figure below shows an example result for "Diabetes_Patient_001."

Deep Learning for Digital Healthcare
Predict Blood Glucose with an LSTM on CGM Data

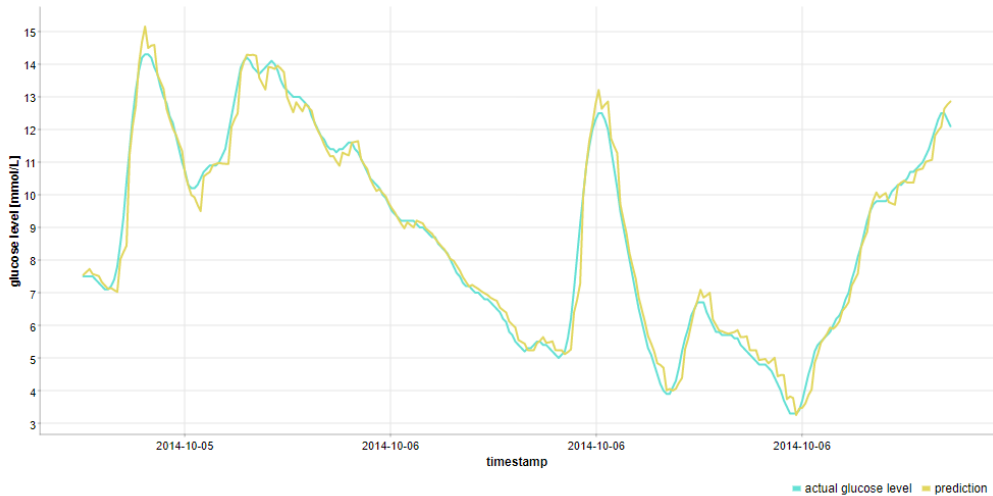
Select Patient

Diabetes_Patient_001

Refresh

Glucose Level Prediction

RMSE score of 0.37



Results for "Diabetes_Patient_1," with an RMSE score of 0.36.

The Root-Mean-Squared-Error (RMSE) score is 0.36, which is pretty good and means that the difference between the actual and predicted data is not that big. This can also be seen in the line plot, showing the actual and the predicted glucose values.

Let's see if the results also look this good by using the validation data. I used "Diabetes_Patient_008" for this. Figure below shows the actual and predicted glucose values for "Diabetes_Patient_008" on October 2nd. You can select different dates to make the view more clear, but the overall RMSE value for the validation data is 0.33. Pretty good!

Deep Learning for Digital Healthcare
Predict Blood Glucose with an LSTM on CGM Data

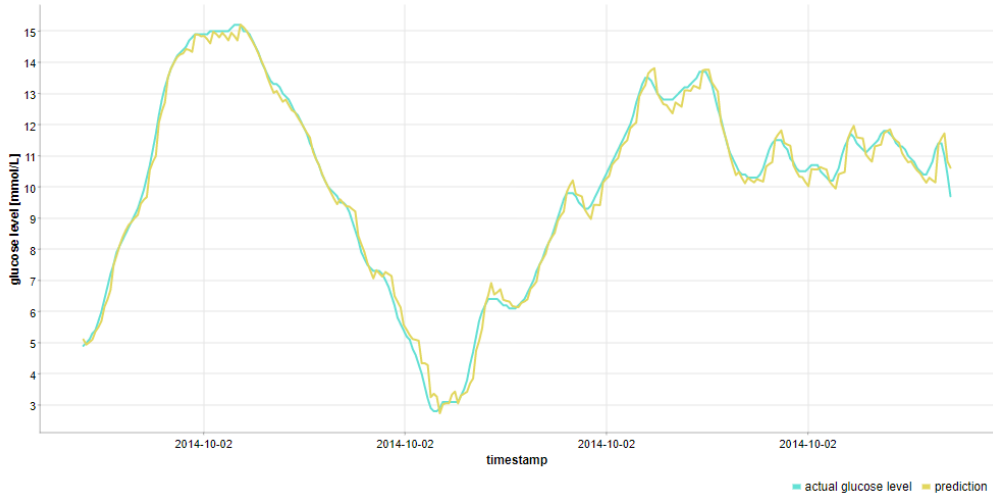
Select Date

2. Oktober

Refresh

Glucose Level Prediction - Diabetes_Patient_008

RMSE score of 0.36



Results for "Diabetes_Patient_8," with an RMSE score of 0.33. In the view, you can select different dates. In this example, I selected October 2nd.

If you want to see more results, check out the workflow and play around with it yourself. Now that it's been trained and tested, the model can be deployed for further purposes.

CGM Data Prediction – The Everyday Helper

CGMs can do a lot to help someone keep track of their blood glucose levels during the day. Future blood glucose predictions can help even more. With this, diabetic people could, for example, estimate whether they'll need insulin in the next 15 minutes. In this post, you saw how a Many-to-Many LSTM network can be used for such predictions. Check out the Digital Healthcare space on the KNIME Hub for more workflows to download and use yourself. Thanks for reading!

Learn how to classify Electrocardiogram Signals with Deep Learning in KNIME

Author: [Ali Asghar Marvi](#), KNIME

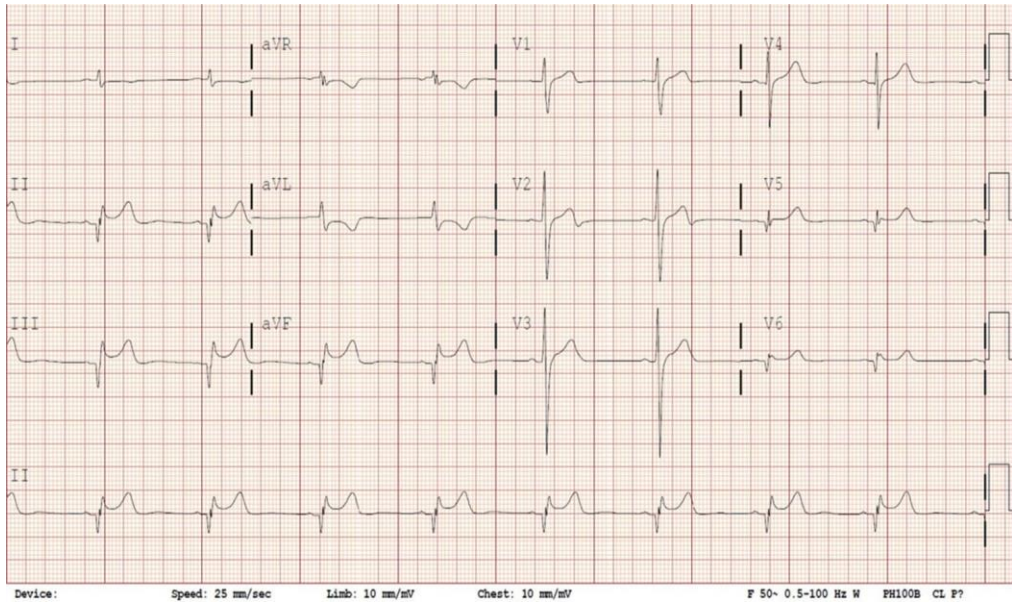
Workflow on the KNIME Community Hub: [ECG Heartbeat Classification Dataset \(PTB\)](#)

Use a Deep Learning Solution in KNIME to Classify Heart Signals

ECG – it's the abbreviated term for [electrocardiogram](#), an [electrogram](#) that records heartbeats. ECG tests are one of the most commonly performed tests to detect heart problems and monitor heart health: Over 100 million are performed annually in the US alone.

The graph produced is a time series of voltage recorded by electrodes placed on the patient's skin. The electrodes detect slight changes in the activity of [cardiac muscle depolarization](#), followed by [repolarization](#) across every cardiac cycle. The changes in signal pattern correspond with various cardiac abnormalities, deficiencies in blood flow through the heart, or electrolyte disbalance. The figure below shows various leads of a sample ECG reading.

In this section, I'd like to give you a walkthrough of an example KNIME workflow that uses deep learning for Electrocardiogram (ECG) classification of normal and abnormal signals. The signals are sampled from [Physionet's ECG Database](#), which is contributed by [Physikalisch-Technische Bundesanstalt \(PTB\)](#). The preprocessed version of the dataset is available on [Kaggle](#). For my example here, I used the files "ptbdb_abnormal.csv" and "ptbdb_normal.csv."

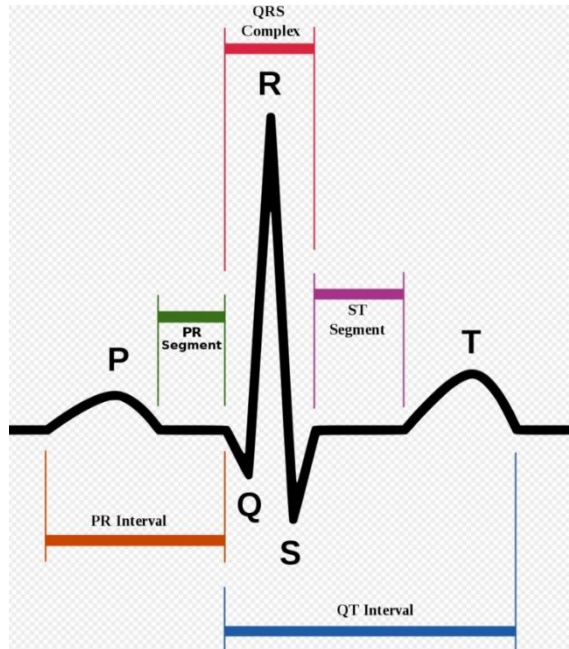


Abnormal [ECG reading](#) in a young patient with shortness of breath.

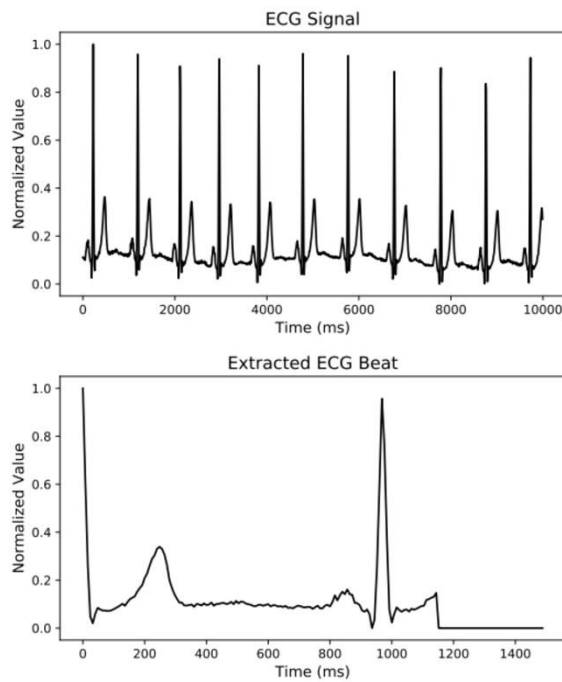
Pre-processing of ECG Signals

Before we dive further into the analysis, I will talk about the pre-processing techniques used by authors Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh to provide complete consolidated CSV files. They applied an effective method for pre-processing signals and extracting beats from them.

To begin, ECG signals are split into 10-second windows, and their amplitude values are normalized between 0 and 1. First derivative of [zero-crossing](#) is applied on each window to find local maximums. Then R-peaks are identified, and a median R-R time interval (τ) is extracted for each window. The R peak is essentially the highest peak of an ECG signal. It is identified using wavelet transformation and is part of the QRS-complex, an oscillation corresponding to contraction and expansion of ventricles and atria, respectively.



[QRS-Complex](#)



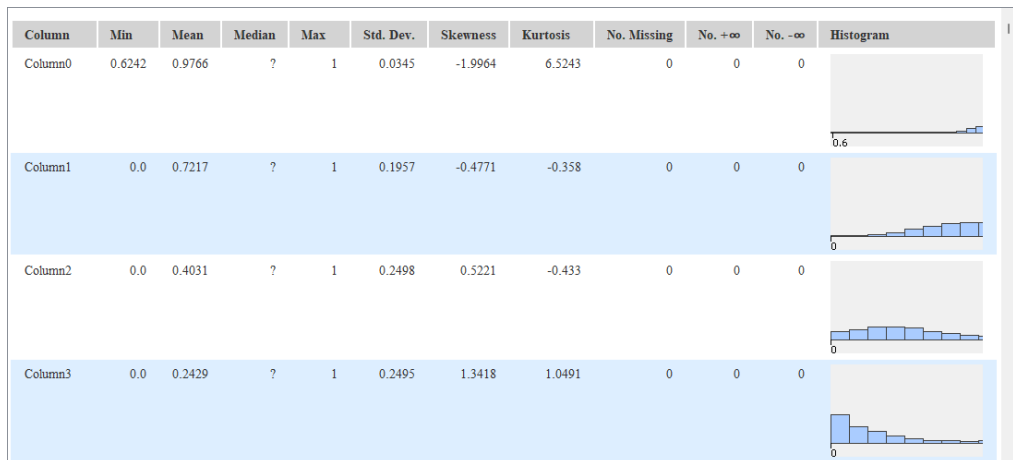
A 10-second sample from the ECG signal and extracted beat from it. Pre-processed output from the paper [“ECG Heartbeat Classification: A Deep Transferable Representation.”](#)

For each R-peak, a signal of length $1.2 T$ is selected and padded with zeros to produce a complete signal of fixed length. The sample data is visualized in figure above.

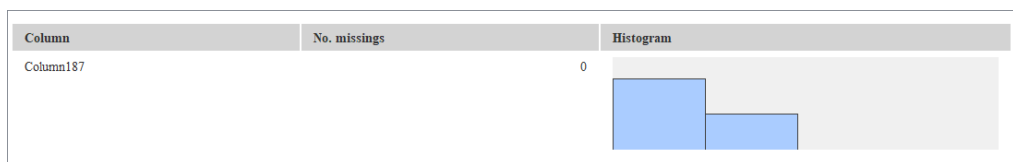
There are two files of PTB ECG datasets provided by the authors. One contains abnormal readings with class variable 1 and is named "ptbdb_abnormal.csv," and the other contains normal readings with class variable 0 and is named "ptbdb_normal.csv." Each file has 188 columns; the last column is the class variable, and the rest represent the signal length, padded with zeros for fixed length. There are no missing values in each column. However, the dataset in total is imbalanced, where abnormal readings are 10,506 and normal readings are 4,046.

Exploratory Data Analysis

Both datasets were concatenated and shuffled into one table. KNIME provides a *Statistics* node in its [KNIME Statistics Nodes](#) Extension. This node is used to describe each column and its respective characteristics. As seen in the figure below, it can be confirmed that there are no missing values in the table. Histograms in each row show the distribution of values in respective columns. Class imbalance is also visible in the bottom part of the figure.

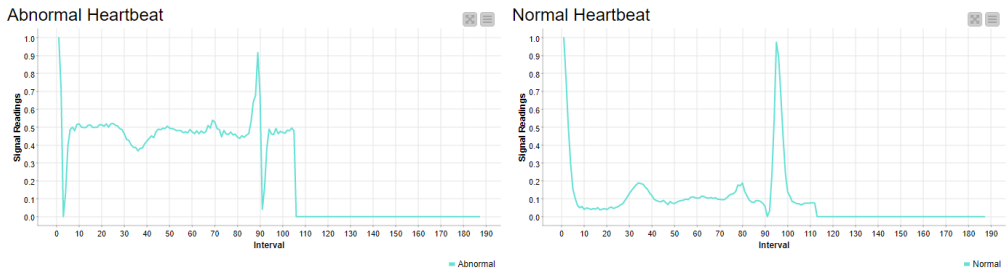


Numerical tab describing statistics of numerical columns.



The Nominal tab describing statistics of class column.

Two beats sampled from each class were also visualized in a line plot, one being the normal and the other the abnormal signal. The abnormal reading has very short peaks and highly fluctuates compared to the normal beat, which is relatively stable and smooth.



Two line plots for sample abnormal (left) and normal (right) reading.

Modeling

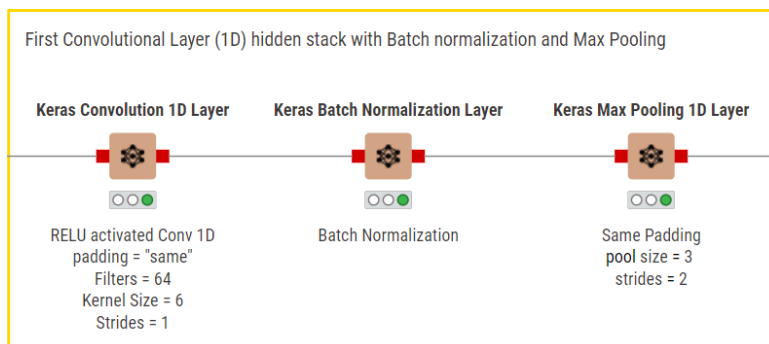
As described in the [research paper](#) by the authors of this dataset, a 1-D Convolutional Neural Network (CNN) architecture is trained to classify ECG beats. The original architecture proposed is quite deep, and took almost two hours to train on an 8GB NVIDIA GPU. For this example, I used a small subset of the proposed architecture with three hidden stacks between input and output layer. The Neural Network was created using the [KNIME Deep Learning – Keras Integration](#).

Deep Neural Network

The input layer had the shape of (1,187), representing the 187 columns and one-dimensional series, i.e. the time series as input and output of two units corresponding to each class.

Hidden Stack 1

As shown in figure below, after the input layer, three hidden layers are added in the first stack. The first layer is a Keras Conv1D Layer using RELU as its activation function, 64

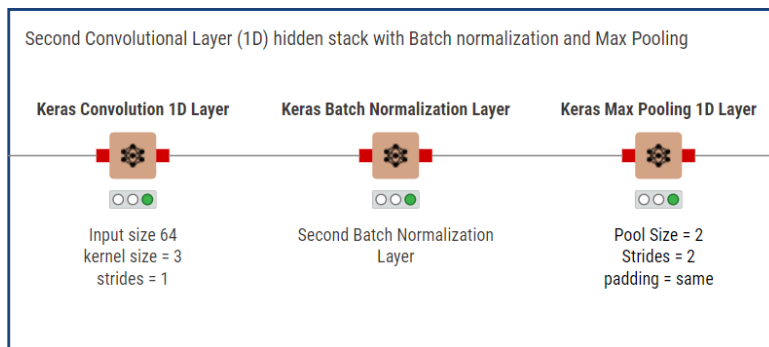


First stack of hidden layers.

filters, a kernel size of 6, strides of 1, and padding set to “same.” It is followed by a *Keras Batch Normalization Layer* node to speed up the training process, then a Keras Max Pooling Layer with pool size of 3 and strides of 2.

Hidden Stack 2

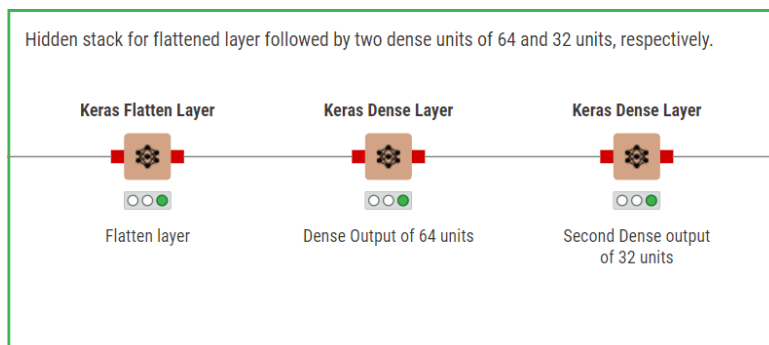
As shown in the next figure, after the first hidden stack in figure above, three hidden layers are added in the second stack. The first layer is a Keras Conv1D Layer using RELU as its activation function, 64 filters, a kernel size of 3, strides of 1, and padding set to “same”. It is followed by a *Keras Batch Normalization Layer* node, then a Keras Max Pooling Layer with pool size of 3 and strides of 2.



Second stack of hidden layers.

Hidden Stack 3

For the third and final hidden stack, no more 1D Convolutional Layers are added. Instead it's a flattened layer, followed by a Dense Layer of 64 units, then a Dense Layer of 32 units before the output layer.

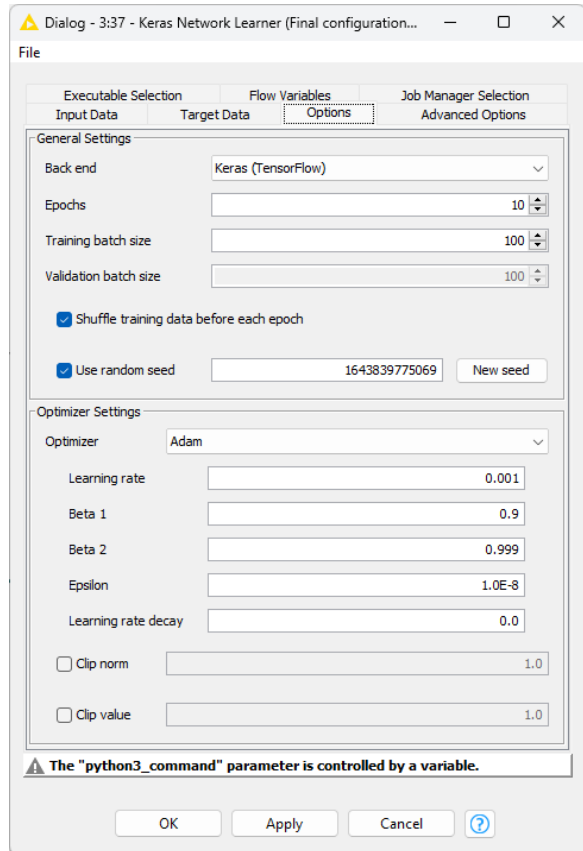


Third stack of hidden layers

Model Training

Before I describe the training process, know that data in KNIME was slightly preprocessed. This includes adjusting class imbalance by oversampling minority class. After that, the class variable was one hot encoded for the model, and the data was partitioned into 80% for training and 20% for testing.

The *Keras Network Learner* node is used to train the model. Since we have a binary classification problem, the loss measure used is “Binary cross entropy”. The model was trained for a total of 10 epochs with an Adam optimizer with relevant parameters. The entire training process took 1 minute and 22 seconds.



Options tab of Keras Network Learner.

Model Scoring

Predictions were made using the *Keras Network Executor* node. Predictions with maximum probability with respect to each class were picked using the *Many to One* node. Finally, the model performance was evaluated using the *Scorer* node. As seen in the next figure, the model performed with the accuracy of 94.7%, with 3,979 rows correctly identified and 224 rows incorrectly identified.

The screenshot shows the 'Confusion Matrix' output from the 'Scorer' node. It displays a confusion matrix table and summary statistics.

Column 187...	0.0	1.0
0.0	2008	94
1.0	95	2006

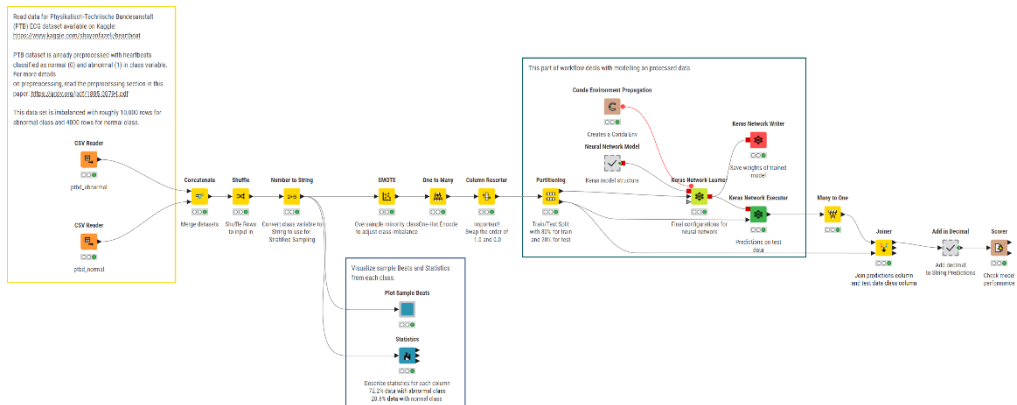
Summary statistics:

- Correct classified: 4,014
- Wrong classified: 189
- Accuracy: 95.503%
- Error: 4.497%
- Cohen's kappa (κ):

Scorer output for model performance.
“0.0” corresponds to normal class, while
“1.0” corresponds to abnormal class.

ECG Classification with KNIME

ECG classification was performed using a Deep Neural Network composed of 1-Dimensional Convolutional Layers, along with Batch Normalization Layers and Max Pooling Layers. The data provided was picked from Kaggle, and was already pre-processed by the authors, so not much needed to be done in that area. The complete workflow can be found on the KNIME Hub in our public Digital Health space. The workflow group comprising the PTB classification on KNIME Hub is “[ECG PTB and MIT-BIH Data Analysis & Modeling](#).” Following figure shows the main workflow of the ECG classification using PTB dataset.



Complete workflow of the ECG - PTB Classification

This part of our analysis catered to the binary classification problem of normal and abnormal beats. In the next part, I will discuss multiclass classification of a different ECG dataset which detects arrhythmia.

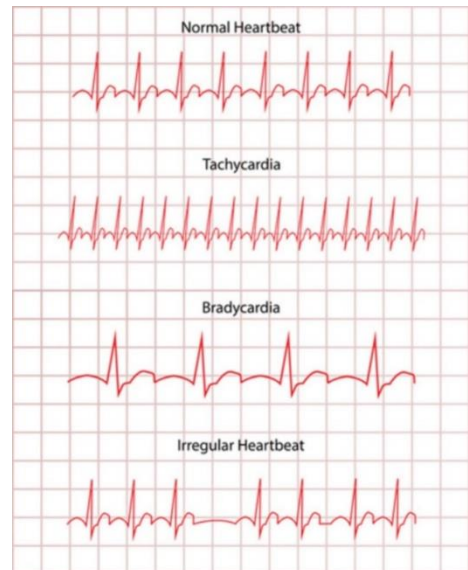
How to perform Electrocardiogram Categorization and Detect Arrhythmia

Author: [Ali Asghar Marvi](#), KNIME

Workflow on the KNIME Community Hub: [ECG MIT-BIH Data Analysis and Modelling](#)

Arrhythmia is a varying rhythm in one's heartbeat. It happens when signals from the brain to the heart are not able to regulate its beat normally. As a result, the heart will beat too quickly ([tachycardia](#)) or slowly ([bradycardia](#)).

This section continues our look at ECG classification with deep learning, using the ECG heartbeat categorization dataset on Kaggle. In this article I want to discuss how to tackle multiclass classification: The dataset, which was compiled and pre-processed from [PhysioNet's MIT-BIH Arrhythmia Database](#), contains five different types of beat categories.



Differentiation between kinds of arrhythmia.

PhysioNet's MIT-BIH Arrhythmia Data

The [ECG heartbeat categorization](#) dataset on Kaggle is composed of two collections of heartbeat signals taken from two famous datasets in heartbeat classification, the MIT-BIH Arrhythmia dataset and the PTB diagnostic ECG database. In the first section about [ECG classification with deep learning](#), I trained the model on the PTB dataset, which has 2 categories of heartbeat signals. In this section, I will be training the model on the MIT-BIH Arrhythmia dataset, which contains 5 different heartbeat signal categories.

In the [Kaggle](#) heartbeat dataset are the “mitbih_train.csv” and “mitbih_test.csv.” among the four files. As the names suggest, there is one file for model training purposes and another for testing purposes. There are 188 columns in each file, just like for the PTB dataset. The target column has five class attributes: 0, 1, 2, 3, and 4. These are their definitions:

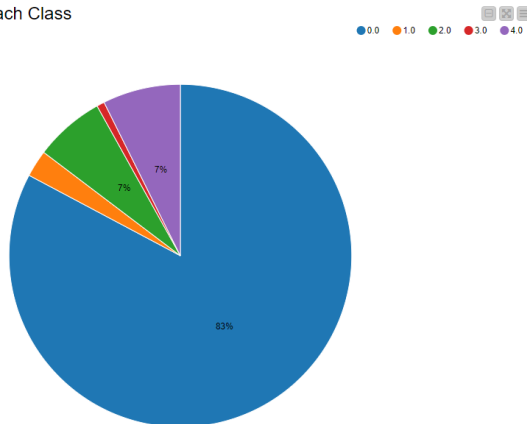
- 0 - “N” for normal heartbeats
- 1 - “S” for supra-ventricular premature
- 2 - “V” for ventricular escape
- 3 - “F” for fusion of ventricular and normal
- 4 - “Q” for unclassified heartbeats

Category	Annotations
N	<ul style="list-style-type: none"> • Normal • Left/Right bundle branch block • Atrial escape • Nodal escape
S	<ul style="list-style-type: none"> • Atrial premature • Aberrant atrial premature • Nodal premature • Supra-ventricular premature
V	<ul style="list-style-type: none"> • Premature ventricular contraction • Ventricular escape
F	<ul style="list-style-type: none"> • Fusion of ventricular and normal
Q	<ul style="list-style-type: none"> • Paced • Fusion of paced and normal • Unclassifiable

Detailed description of all class variables in the MIT-BIH arrhythmia dataset.

Details for each of the classes can be found in the following figure, with the descriptions from the associated [research paper](#) by the authors.

Number of Rows for Each Class



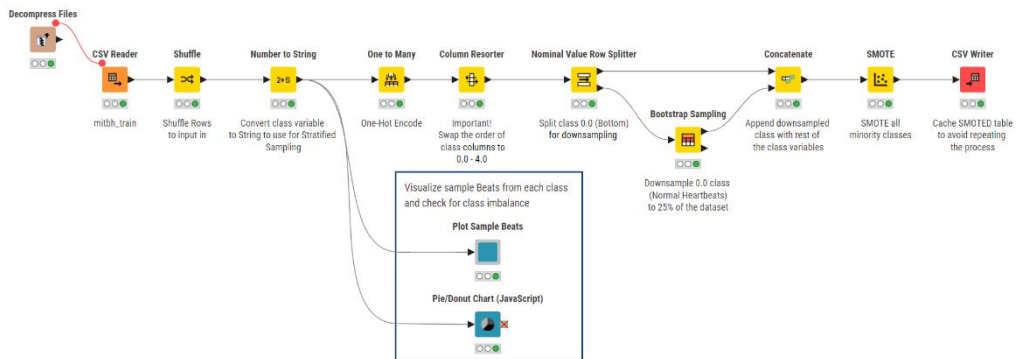
Class distribution in training set. Imbalance is evident here.

The training set has 87,554 rows, while the test set has 21,892 rows. The train set is highly imbalanced, with normal heartbeats labeled “0.0” (83% as shown in following figure). Next, I’ll discuss my data preparation approach to train an unbiased model.

Treating class imbalance for multiple classes

For an unbiased model, class variables were shuffled and converted to string. I visualized the distribution of rows in the training set (the blue annotation block in figure below). Class variables are one-hot encoded and split such that the bottom port outputs all rows corresponding to “normal” class and rest in the top port. Using the *Bootstrap Sampling* node, the “normal” class is down-sampled to return 25% of total normal heartbeats in the dataset. The down-sampled table is concatenated with the rest of the rows belonging to other classes. The resulting table is manipulated using the *SMOTE* node by oversampling minority classes.

Using SMOTE on a large dataset with multiple class variables is a time-consuming process. To make it relatively faster, down-sampling of the majority class is carried out before oversampling. The final output now has equal proportions of rows. *CSV Writer* is used to write the updated table in the data area of the workflow to avoid repeating the process and save time.



Downsampling the majority class and using SMOTE to oversample the minority classes.

Model Training

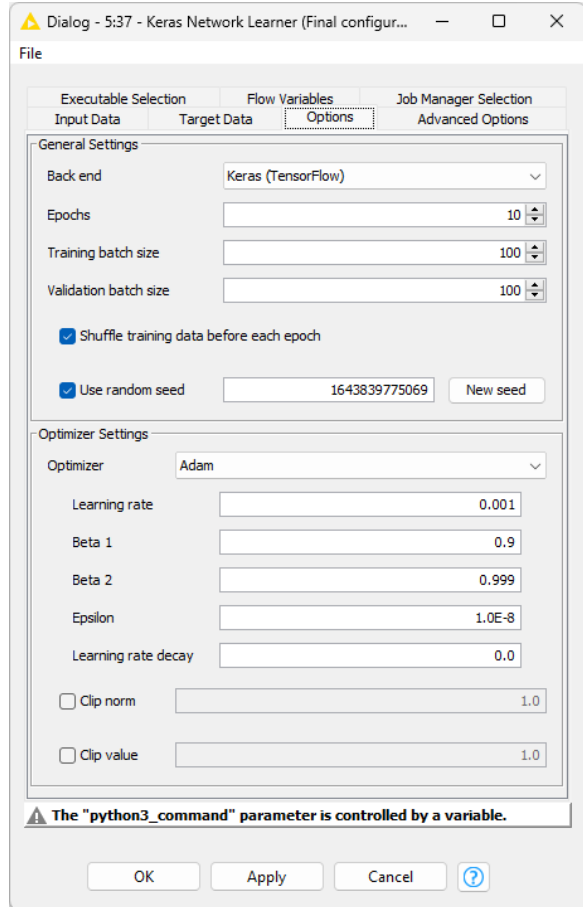
Now that my data is resampled, I will proceed with the data modeling. Training data is partitioned with 80/20 split. Since test data is already provided, 20% of partitioned data was used for validation on each epoch by *Keras Network Learner* node.

The neural network used is the same as in Part 1. However, since it's a multiclass problem, the loss function used here is “Categorical cross entropy”, and it's optimized

using Adam in the *Keras Network Learner* node. The rest of the optimizer settings can be seen in the following figure. Training the model took around 5 minutes and 37 seconds, with a validation accuracy of approximately 97%.

Model Scoring

Similarly to the workflow from the previous chapter, a *Keras Network Executor* node is used to make predictions on the test set provided by the authors. The *Many to One* node is used to extract predictions with maximum probability, while model performance is measured using the *Scorer* node. As seen in the figure below, the model makes predictions with 94.1% accuracy, with 20,617 rows correctly identified and 1,275 wrongly classified.



Configuration window of Keras Network Learner node.

The image shows the 'Confusion Matrix - 5:74 - Scorer (Check model)' window. It displays a confusion matrix table with columns for predicted classes (4.0, 3.0, 2.0, 1.0, 0.0) and rows for actual classes (4.0, 3.0, 2.0, 1.0, 0.0). Below the table, it shows summary statistics: 'Correct classified: 20,566', 'Wrong classified: 1,326', 'Accuracy: 93.943%', 'Error: 6.057%', and 'Cohen's kappa (κ): 0.821%'.

Column187...	4.0	3.0	2.0	1.0	0.0
4.0	1571	4	11	6	16
3.0	0	142	8	0	12
2.0	5	35	1355	7	46
1.0	2	4	12	457	81
0.0	61	261	189	566	17041

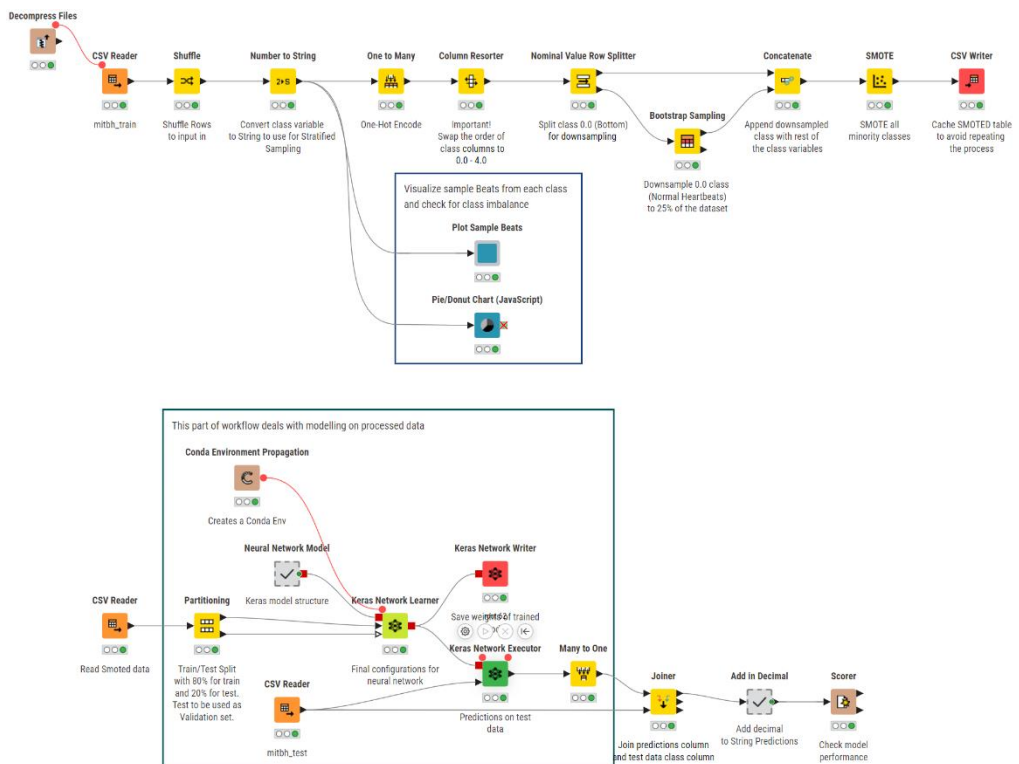
Correct classified: 20,566 Wrong classified: 1,326
Accuracy: 93.943% Error: 6.057%
Cohen's kappa (κ): 0.821%

Confusion Matrix of Model performance.

Conclusion

In this second part of ECG classification, I have discussed an arrhythmia dataset, which is available on Kaggle. The dataset was highly imbalanced, with the majority class being "Normal Heartbeat." The imbalance was adjusted using Bootstrap Sampling (downsampling) of the majority class and oversampling of the minority class using the SMOTE technique.

Similarly to before, data was trained using 1-Dimensional Convolutional Neural Network architecture, including Batch Normalization and Max Pooling. The model performed with a higher accuracy on the test set. The complete workflow can be found on the KNIME Hub under the Digital Health public space.



Complete workflow of ECG arrhythmia classification in KNIME.

The workflow group comprising the ECG arrhythmia classification on the KNIME Hub is [ECG PTB and MIT-BIH Data Analysis & Modeling](#), under the name "ecg_cnn_mit." The figure above shows the main workflow of the ECG classification using the arrhythmia dataset.

Leveraging Healthcare Literature

This chapter highlights applications that require the extraction of information from biomedical literature, such as keyword search or categorization through ontologies. With KNIME Analytics Platform's dedicated extension for text processing, users can apply natural language processing (NLP) techniques to pre-process text and extract meaningful information from unstructured data. These techniques and visual workflows enable users to easily blend data from structured sources (see Chapter 2) with information from text.

This chapter includes the articles:

- **Tagging Disease Names in Biomedical Literature**, p. 73
 - Jeany Prinz, *KNIME*
- **Improve Literature Search & minimize Information Overload**, p. 81
 - Dayanjan Wijesinghe, *Virginia Commonwealth University*
 - Martyna Pawletta, *KNIME*

Tagging Disease Names in Biomedical Literature

Author: [Jeany Prinz](#), KNIME

Workflow on the KNIME Community Hub: [Fun with Tags](#)

Introduction

The rapid growth in the amount of biomedical literature becoming available makes it impossible for humans alone to extract and exhaust all of the useful information it contains. There is simply too much there. Despite our best efforts, many things would fall through the cracks, including valuable disease-related information.

Hence, automated access to disease information is an important goal of text-mining efforts¹². This enables, for example, the integration with other data types and the generation of new hypotheses by combining facts that have been extracted from several sources¹³.

In this blog post, we will use [KNIME Analytics Platform](#) to create a model that learns disease names in a set of documents from the biomedical literature. The model has two inputs: an initial list of disease names and the documents. Our goal is to create a model that can tag disease names that are part of our input as well as novel disease names. Hence, one important aspect of this project is that our model should be able to autonomously detect disease names that were not part of the training.

To do this, we will automatically extract abstracts from [PubMed](#) and use these documents (the corpus) to train our model starting with an initial list of disease names (the dictionary). We then evaluate the resulting model using documents that were not part of the training. Additionally, we test whether the model can extract new information by comparing the detected disease names to our initial dictionary.

Subsequently, we interactively inspect the diseases that co-occur in the same documents and explore genetic information associated with these diseases.

Our workflow has three main parts, which will be described in detail in the following:

¹² (2013, August 21). *DNorm: disease name normalization with pairwise learning to rank*. Retrieved July 12, 2018, from <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3810844/>

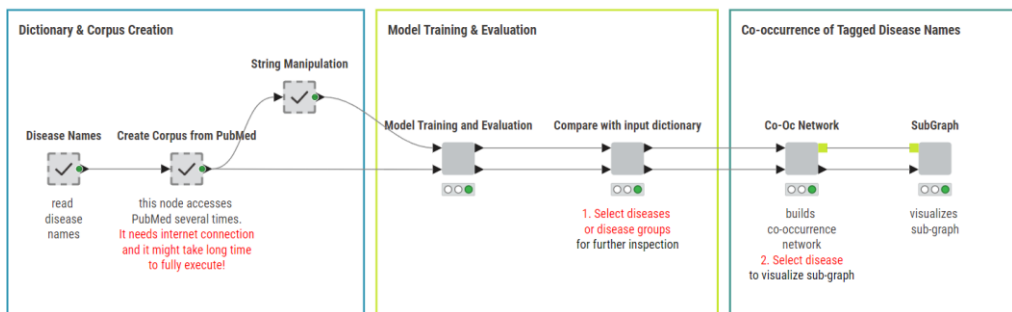
¹³ (2006, February 1). *Literature mining for the biologist: from information retrieval to ...* - *Nature*. Retrieved July 12, 2018, from <https://www.nature.com/articles/nrg1768>

1. Dictionary and Corpus Creation
2. Model Training and Evaluation
3. Co-occurrence of Tagged Disease Names

Fun with Tags

Automated access to disease information is an important goal of information extraction and text mining efforts. Here, we want to create a model that learns disease names in a set of documents from biomedical literature. We will automatically extract literature from PubMed and use these documents to train our model on an initial set of disease names (the dictionary). We score the resulting model and check if we can extract new information by comparing the detected disease names to our initial set. Subsequently, we interactively inspect the diseases that co-occur in the same documents by a network approach and look into genetic information associated with these diseases.

To learn more about this workflow, please read: <https://www.knime.com/blog/fun-with-tags>



Overview of the workflow to automatically extract disease related information from biomedical literature. First, the literature corpus as well as the dictionary of known disease names are gathered. Next, the model is trained and evaluated. Last, the results are investigated in a network graph.

1. Dictionary and Corpus Creation

Dictionary creation (Disease Names)

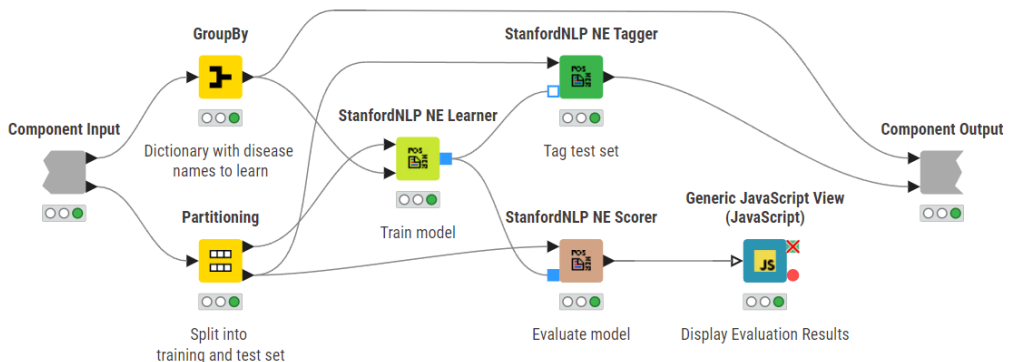
For the initial input, we create a dictionary that contains disease names from [Ensembl](#) [Biomart](#). For that, we downloaded phenotypes (diseases and traits) that are associated to genes or variants. These diseases and traits are assembled from different sources such as [OMIM](#), [Orphanet](#), and [DDG2P](#). To create a dictionary that contains commonly used disease names, we filter the documents for disease names that are contained in at least three sources. The model we eventually train is case sensitive, thus, we add variations such as capitalizing the disease names, lowercase, and uppercase. The resulting disease names comprise our initial input dictionary. The dictionary creation is contained in the *Disease Names* metanode in the main workflow.

Corpus creation

One of the most important steps for creating a NLP (natural language processing) model is to gather a corpus of documents on which to train and test the model. For our purpose of automatically accessing disease information, we use abstracts from the database [PubMed](#). The *Document Grabber* node enables us to automatically search the PubMed database according to specific queries. This query takes a disease name from our dictionary and searches for it in the PubMed data. We only keep the results from diseases with at least 20 hits in PubMed, and we collect a maximum of 100 documents per disease. This Corpus is created in the *Create Corpus from PubMed* metanode.

2. Model Training and Evaluation

Model



Workflow contained in the wrapped metanode "Model Training and Evaluation". We use the StanfordNLP NE Learner and Tagger nodes to tag disease names in our corpus. The evaluation is done using the StanfordNLP NE Scorer and displayed using a Generic JavaScript View.

We can now use the dictionary and our corpus as input for the *StanfordNLP NE Learner* node. The *StanfordNLP NE Learner* node creates a Conditional Random Field (CRF) model based on documents and entities in the dictionary that occur in the documents. CRFs, a type of sequence model, which takes context into account, are often applied in text mining. If you are interested in the StanfordNLP toolkit, please visit <http://nlp.stanford.edu/software/>.

The figure above depicts the workflow contained in the wrapped metanode *Model Training and Evaluation*. As the figure shows, we first split our collected documents into a training (10%) and a test set (90%) and train the model using the training data. We use the default parameters, with the exception that we increase maxLeft (the maximum context of class features used) to two and Max NGram Length (maximum

length for n-grams to be used) to ten. Additionally, we select the Word Shape function [dan2bio](#).

Next, we tag the documents in our test data with our trained model. Subsequently, we use the same test data to score our model. This is done with the *StanfordNLP NE Scorer* node, which calculates quality measures like precision, recall, and F1-measures and counts the amount of true positives, false negatives, and false positives. Note that it does not make sense to calculate true negatives, as this would be every word that is correctly *not* tagged as a disease. Internally, the *StanfordNLP NE Scorer* node tags the incoming test document set with a dictionary tagger using our initial disease dictionary. After that, the documents are tagged again via the input model, and then the node calculates the differences between the tags created by the dictionary tagger and the tags created by the input model.

Confusion Matrix

Actual	Predicted	
	positive	negative
positive	8212	743
negative	287	xx

Performance:

Precision: 0.966
Recall: 0.917
F1: 0.941

Confusion matrix containing the true positives, false positives, and false negatives. Precision, Recall and F1 are also shown.

The *Generic JavaScript View* node helps us to generate a view summarizing the results. As can be seen in figure below, we achieve a Precision of 0.966, Recall of 0.917, F1 of 0.941.

Comparison with input dictionary

Now comes the interesting part. We are not only interested in how well our model recaptures disease names that we already know, but also if we are able to find new disease names. Therefore, we divide the diseases we have found in the test set depending on whether or not they were in our initial dictionary. We flag these either as “disease name contained in the input dictionary” or, alternatively, as “disease name NOT contained in the input dictionary.”

We then create an interactive view that allows the user to investigate and filter the results. To select all data that were(or were not) contained in the input we use a *GroupBy* node to group according to the attribute we just created (i.e., if the data were part of the input dictionary or not). Here we use a small trick: it is important to “Enable highlighting” in the *GroupBy* node. If we show that in a composite view using a Table View (JavaScript) alongside another *JavaScript* view, it is now possible to make selections in one view which affect the other as well. If the user does not select anything, we will use all diseases by default.

This part is included in the metanode named *Compare with input dictionary*.

Tagged Diseases

Select diseases of interest that will be inspected in next nodes/views

You can select groups of diseases (occurrences where the disease name is either contained or not contained in the input dictionary) or individual disease names. All disease names can be selected by clicking the check boxes in the upper left corner of each table.

Show entries

Search:

<input checked="" type="checkbox"/>	data contained in	
<input checked="" type="checkbox"/>	disease name contained in the input dictionary	
<input checked="" type="checkbox"/>	disease name not contained in the input dictionary	

Showing 1 to 2 of 2 entries

Previous **1** Next

Show entries

Search:

<input checked="" type="checkbox"/>	Term as String		<input checked="" type="checkbox"/>	data contained in	
<input checked="" type="checkbox"/>	WEAVER OVERGROWTH SYNDROME			disease name not contained in the input dictionary	
<input checked="" type="checkbox"/>	WEAVER SYNDROME			disease name contained in the input dictionary	
<input checked="" type="checkbox"/>	WEAVER-LIKE SYNDROME			disease name not contained in the input dictionary	
<input checked="" type="checkbox"/>	WEYERS ACROFACIAL DYSOSTOSIS			disease name not contained in the input dictionary	

Interactive view of the results. The user can select one or more diseases or even all diseases that were (not) part of the input in the lower table view. This affects the first table and shows the corresponding diseases appearing in the test set. This is the output of the metanode "Compare with input dictionary".

The disease names detected in the test set that are not contained in the input dictionary can be very similar to the ones that we used for the training. For example, PYCNODYSOSTOSIS is contained in our input dictionary and we detect the new name PYCNODYSOSTOSIS SYNDROME as well as the misspelling PYCNODYSOTOSIS, which we flag as not part of the input dictionary. The similarity of the tags shows us that the tagged disease names that are not part of the input dictionary actually do make sense. These alternate tags can be valuable, for example, in normalization efforts where we need to determine synonyms and/or spelling variants of disease names.

To learn more about the newly detected disease names where the relationship to our input is less clear, we investigate whether or not tagged disease names co-occur in the same documents. This enables us to, for example, infer information from known diseases to the ones that were not in the input dictionary.

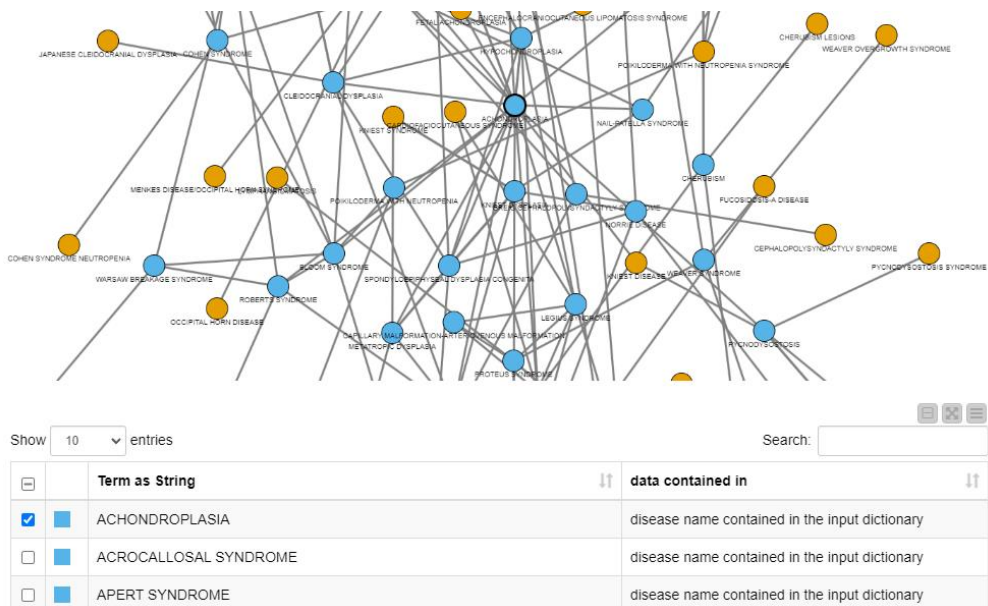
3. Co-occurrence of Tagged Disease Names

In this last part, we utilize the *Term Co-Occurrence Counter* node to count the number of co-occurrences for the list of tagged diseases within the documents. We add the information to the resulting disease pairs pertaining to whether or not each term was part of the input dictionary.

Co-occurrence network

To facilitate the investigation of the results, we created a network graph with diseases as nodes, which were connected if they co-occurred in the same document. We colored the nodes according to their flag specifying whether or not they were contained in the input dictionary. We then created a view containing the network as well as a table with the disease names and the annotation stating whether it was part of the input dictionary. The creation of the network graph, node assignment, coloring, and edge definition is all computed in the metanode named *Co-oc Network*. The view is shown in the next figure.

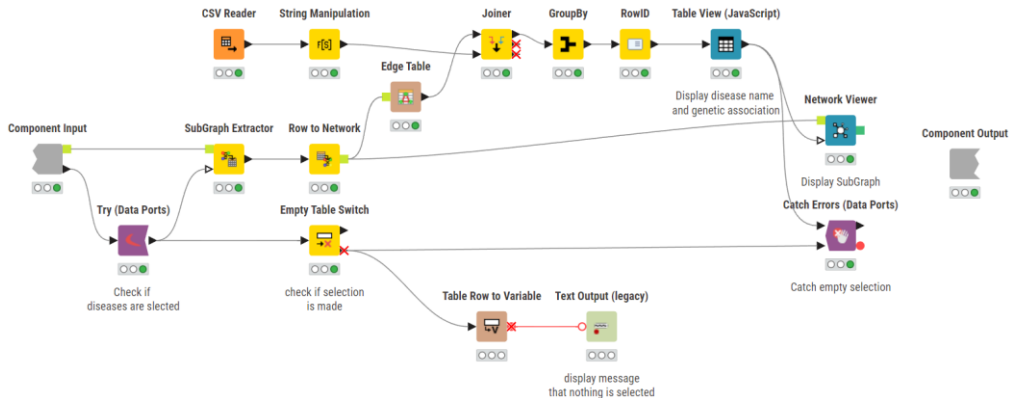
The user can now select nodes or rows of interest for further inspection either in the network or in the table. The subgraph surrounding the selected nodes/rows will be extracted and displayed in the next metanode.



Network view of co-occurring disease names. Each node is a disease names. Nodes are connected if the disease name co-occur at least once in a document. The node color refers to the presence (blue) or absence (yellow) of the disease name in the original dictionary.

Subgraph

The *SubGraph Extractor* node enables us to focus on a specific subset of diseases and their neighbors in the co-occurrence network. For that, the user needs to select a disease of interest. If nothing is selected, we display a message stating that one did not select a disease to inspect in subgraph. We utilize the *Try* and *Catch Errors* nodes to check if the selection is empty.



Workflow to extract subgraphs of interest in the network of co-occurring disease names. This workflow is contained in the last metanode, "SubGraph".

Again, we show the network (*Network Viewer* node) in an interactive view along with a table (*Table View* node) containing the disease names. Furthermore, we display additional information about genetic associations of the disease in the same table. We do this by joining the edge table of our network with genetic information about the diseases that we collected from Ensembl Biomart. This allows us to derive hypotheses about the genetic basis of the diseases that were not in our input dictionary.

For example, we select OHDO SYNDROME, which was not part of our input dictionary. In the resulting view, we see that this disease co-occurs with GENITOPATELLAR SYNDROME. Using the Ensembl information, we know that GENITOPATELLAR SYNDROME is associated with the gene, KAT6B. This could lead to the working hypothesis that OHDO SYNDROME is also associated with KAT6B. Indeed, mutations in the KAT6B gene have been associated with the Say-Barber-Biesecker Variant of Ohdo Syndrome¹⁴.

¹⁴ "Whole-exome-sequencing identifies mutations in histone ... - NCBI." 11 Nov. 2011, <https://www.ncbi.nlm.nih.gov/pubmed/22077973>. Accessed 1 Aug. 2018.

Leveraging Healthcare Literature Tagging Disease Names in Biomedical Literature



Show entries Search:

Node id	Gene name
GENITOPATELLAR SYNDROME	KAT6B
NAIL-PATELLA SYNDROME	LMX1B
OHDO SYNDROME	?

Subgraph connecting GENITOPATELLAR SYNDROME, OHDO SYNDROME and NAIL-PATELLA SYNDROME. Blue nodes indicate that the disease was part of our input dictionary, whereas yellow nodes indicate that the disease was not included.

Summary

Today, we successfully trained a model to tag disease names in biomedical abstracts from [PubMed](#). We started with a set of well-known disease names in a dictionary. We then interactively investigated these known diseases as well as diseases that were not in our original dictionary and checked their co-occurrence in the collected documents. From these co-occurrences, we created a co-occurrence network where we could easily zoom into connected subgraphs and their underlying genetic associations.

In summary, we learned how to tag new and known disease names in KNIME Analytics Platform, and hopefully you had fun!

Improve Literature Search & minimize Information Overload

Authors: [Dayanjan Wijesinghe](#), Virginia Commonwealth University & [Martyna Pawletta](#), KNIME

Workflows on the KNIME Community Hub: [Search for Ethnicity Related Adverse Events of a Drug](#) & [Tagging Genes in Disease Related Publications](#)

Examples of Tagging Scientific Publications using Synonyms, Dictionaries & Ontologies

An ever increasing body of scientific literature is documenting significant advances in healthcare research and novel treatments. However, these valuable insights often get lost in the exponentially increasing volume of published studies.

A single query often results in a huge number of publications, all of which needs to be carefully read through. This quickly becomes an overwhelming task. Narrowing the query parameters to bring down the resultant information to a manageable number runs the risk of the query being too specific and leading to a few or even no results. These narrow queries can also be a significant barrier to incidental findings that may be of relevance to the question under investigation.

A simple approach allowing the querying of two overlapping but different sources such as [PubMed](#) or [Semantic Scholar](#) in parallel, and using highlighting to quickly identify the relevant information will not only save time, but go a long way towards minimizing information overload while keeping a query broad enough to allow incidental findings.

In today's blog we will demonstrate how such an outcome may be achieved using the open source KNIME Analytics Platform.

We created two example workflows to automatically extract abstracts from publications.

- Example #1 investigates Adverse Drug Events (ADE's) in the context of the ethnic background of a patient: Download workflow: [Search for Ethnicity Related Adverse Events of a Drug](#).
- Example #2 investigates relationships of particular genes and diseases. Download workflow: [Tagging Genes in Disease Related Publications](#)

With both workflows we use ontologies, dictionaries, or web services to get a list of terms or synonyms to help with the search. We also tag terms of relevance in abstracts

and highlight them in a manner to make the relevant information much easier to find. Let's have a look at the two workflows.

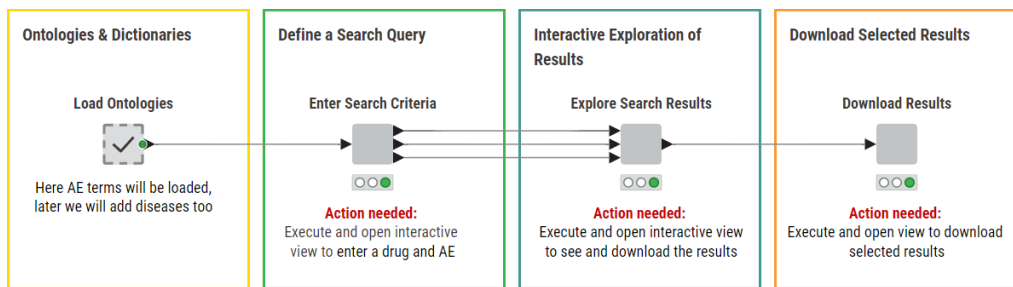
Example #1: Investigating Adverse Drug Events & Ethnicity

Genetic polymorphisms in enzymes involved in ADME (absorbance, distribution, metabolism and excretion) have been demonstrated to affect drug response as well as ADE's. Specific polymorphisms in these enzymes have been demonstrated to be differentially concentrated among different ethnic groups. This natural phenomenon leads to ethnicity dependent variations in drug responses and ADEs.

Often, the balanced representation of different ethnic groups in clinical trials are difficult to achieve. Thus medications once approved are used to treat ethnic groups on which the medication are not tested. As such, a drug that is safe for the majority may have a differential response or ADE in a different ethnic group due to inherent variations in polymorphisms among ADME genes.

Early identification of such rare adverse events will likely be detailed in clinical reports with one or a small number of patients. Thus such information is often lost in the vastness of published literature unless a routine ADE surveillance effort is implemented that specifically investigate ADE's in the context of ethnic backgrounds.

In the workflow we have created, we look into the example of [Carbamazepine](#) and the [Stevens-Johnson Syndrome](#) which is a serious skin disorder. Let's check if there is literature available that would suggest a dependency on any kind of ethnic background of patients.



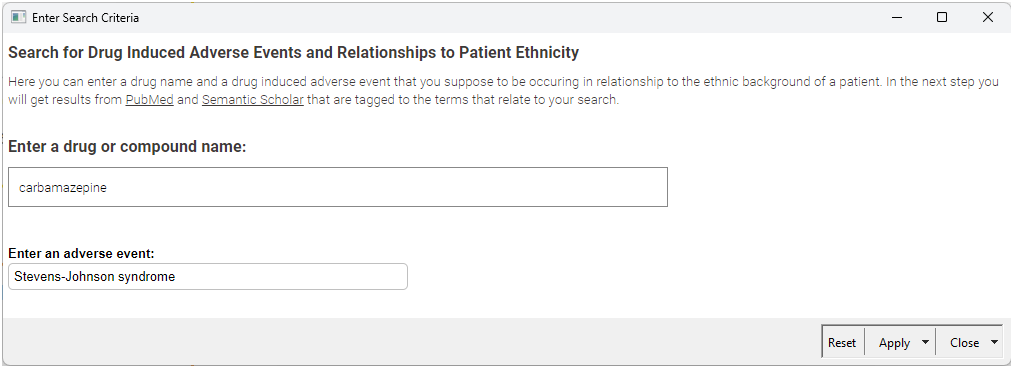
This example workflow gets an adverse event and a drug from the user and gets tagged publications that mention a relationship to any kind of ethnicity.

Create Query for Web Services

With this outline in mind we created a workflow that allows the user to search within two different literature sources for publications that relate to the search query using only one tool – the open source KNIME Analytics Platform.

In the first step the user can add a name of a drug and an adverse event into an interactive view of a [component](#). It's not only two simple text fields – we added more functionality in the backend that to improve the search. The adverse event field is made of an [Autocomplete Widget](#) node and, based on the [Ontology of Adverse Events](#), proposes terms directly when the user is typing first characters. Nevertheless, if a term isn't available, it will use what is entered to create the query.

Depending on the author of a publication, different names of a particular drug that is described is used. Sometimes the compound name or the marketed drug name could be entered. For example: some people would say Aspirin, others Acetylsalicylic Acid. To make sure we catch as many publications as possible, we collect synonyms for the entered drug using the [ChEMBL Web Service](#).



Interactive View of the Enter Search Criteria component that allows the user to specify the search criteria.

Finally, in order to improve the search, we added two different sources to look for publications. The first is PubMed that can be called using the [Document Grabber](#) node.

In our research to see if other APIs are available, we stumbled upon [Semantic Scholar](#) which is an invention of the Allan Institute and contains a really easy to use web service. Please read through the [license](#) and the [API documentation](#), before you start using it in the workflow.

Remember the time when publications were printed and you used to sit with a text highlighter or marker to mark the important parts? This is done here automatically on the abstracts using the [Dictionary Tagger](#) and [Tagged Document Viewer](#) nodes.

Explore Search Results

Show: 5 entries

Search:

Title	url
<input type="checkbox"/> Potential role of regulatory DNA variants in modifying the risk of severe cutaneous reactions induced by aromatic anti-seizure medications .	PubMed
<input type="checkbox"/> Genetic Markers for Stevens-Johnson Syndrome/Toxic Epidermal Necrolysis in the Asian Indian Population : Implications on Prevention .	PubMed
<input type="checkbox"/> Human Leukocyte Antigen Gene Testing and Carbamazepine-Induced Toxic Epidermal Necrolysis : A Study of Pediatric Practice .	PubMed
<input checked="" type="checkbox"/> The HLA-B*15 :02 polymorphism and Tegretol®-induced serious cutaneous reactions in epilepsy : An updated systematic review and meta-analysis .	PubMed
<input type="checkbox"/> HLA Association with Drug-Induced Adverse Reactions .	PubMed

Showing 1 to 5 of 100 entries

Previous 1 2 3 4 5 ... 20 Next

Show: 1 entries

☒ **The HLA-B*15 :02 polymorphism and Tegretol®-induced serious cutaneous reactions in epilepsy : An updated systematic review and meta-analysis .**

The HLA-B*15 :02 polymorphism and Tegretol®-induced serious cutaneous reactions in epilepsy : An updated systematic review and meta-analysis .

Drug Tegretol® [carbamazepine (CBZ)], an aromatic drug approved for epilepsy treatment , can induce adverse drug reactions (ADRs) after its administration . Several genetic studies of epilepsy have shown that genetic polymorphisms increase the risk of ADRs , and some interactions between CBZ and other treatments can also induce adverse effects . Thus , to avoid such interactions and to provide an overview of the genetic profiles involved in ADRs with CBZ , for the first time , a systematic review and meta-analysis focusing on epilepsy was performed , using Cochrane Library , Embase and PubMed databases to find studies published between January 1980 and October 2016 . Of the eligible studies , those selected were related to the impact of genetic polymorphisms on ADRs in patients receiving antiepileptic treatment . The results of these selected studies are expressed as pooled odds ratios (ORs) with 95 % confidence intervals (CIs) , based on data from individual patients . Out of 807 articles , nine were included in the present meta-analysis to assess the association between human leukocyte antigen (HLA)-B*15 :02 polymorphisms and CBZ-induced serious cutaneous reactions (SCRs) , such as

Adverse_event Stevens-Johnson syndrome (SJS) and toxic epidermal necrolysis (TEN) , in epilepsy . It was found that HLA-B*15 :02 polymorphisms were significantly

associated with CBZ SCR risk (OR : 27.325 , 95 % CI : 9.933-51.166) , while subgroup analyses by **Ethnicity** showed that the association was significant in

Ethnicity Han Chinese (OR : 42.059 , 95 % CI : 9.587-184.514) . The HLA-B*15 :02 polymorphism was also strongly associated with the CBZ-SJS/TEN subgroup (OR : 152.089 , 95 % CI : 34.737-665.901) and significantly associated with the CBZ-SJS/TEN subgroup (OR : 13.993 , 95 % CI : 7.291-26.856) . Also , the

Reset Apply Close

This example workflow gets an adverse event and a drug from the user and gets tagged publications that mention a relationship to any kind of ethnicity

Check the Results

So let's have a look at our example Carbamazepine & Stevens-Johnson Syndrome to see if there is literature available that shows a relationship between both terms and the ethnic background of patients. Indeed there is a lot of information already available (see figure above).

Now it's the time to go through those and check for the interesting ones. Once this is done, a list of relevant publications can be generated and downloaded using the interactive view of the last component in the workflow.

Note. This workflow uploaded to [KNIME Business Hub](#) creates a web-based application that doesn't expect any workflow building skills from the user."

Example #2: Investigating Relationships of Specific Diseases & Genes

With this example we will show how easy it is to use dictionaries and ontologies and tag those terms on biomedical literature within KNIME Analytics Platform. It is in principle a similar example to the first one – we aim to do a literature search – but let's focus here on the reading part a bit more.

We start again with an input - this time a disease that needs to be entered or selected in the [Autocomplete Widget](#) node. The node gets a list of diseases from the disease ontology that can be downloaded from the OBO foundry. To read the .rdf format we can use the [Triple File Reader](#) node.

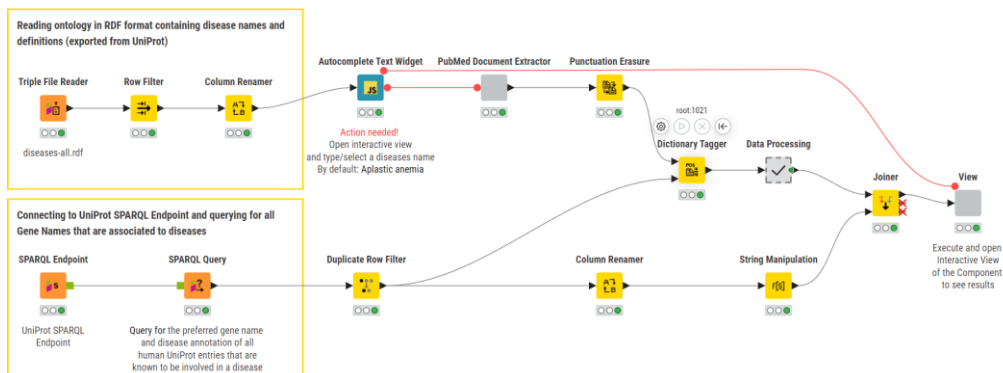
The literature that will be downloaded with title and abstract will be in the next step tagged with genes to find out which genes are mentioned in publications with a certain disease. For this we connect with the [UniProt SPARQL Endpoint](#) using the [SPARQL Endpoint](#) node. With a dedicated query we get a list of genes and disease annotations from UniProt.

Document Tagging Using Ontologies and Dictionaries

This example workflow shows how ontology terms can be used to tag biomedical literature.

In the first step, the *Triple File Reader* node reads an ontology in RDF format (extracted from UniProt) and allows the user to select a disease (using the *Autocomplete Text Widget* node). Then, abstracts from PubMed for the specified disease are automatically extracted. Additionally, a connection to the UniProt SPARQL Endpoint is made and a SPARQL Query executed that allows to extract preferred gene names and disease annotations of all human UniProt entries that are known to be involved in a disease. The gene names are used as the input for the *Dictionary Tagger* together with the extracted documents from PubMed. In the last step a component allows to inspect the tagged data.

Note: To open the interactive view of the "View" component, click the magnifier glass in the component's tool bar.



This example workflow combines different ontologies to extract information from PubMed and adds Tags in an interactive view at the end.

Once the terms and literature is loaded we map the genes against the publications using the *Dictionary Tagger* node. With some more processing and data preparation we create a visualization within a component that allows us to interactively browse through the publication.

To make it easier to browse through the genes we counted the tags and those with the highest frequency are visible first. With this the user can decide if the known and

already well researched, or the rare and maybe newly discovered genes are of interest for the disease under investigation.

View

Genes related to: Aplastic anemia

Those gene names were tagged in the list of document for your selected diseases

Show entries

Hr

Term Frequency: 676

☐

Fancd2

Term Frequency: 625

☐

Tp53

Term Frequency: 576

☒

Fanca

Term Frequency: 529

☐

Tpo

Term Frequency: 288

☐

Showing 1 to 5 of 171 entries

Previous

1

2

3

4

5

...

35

Next

UniProt Annotation

Show entries

Search:

UniProt Disease Annotation

Disease susceptibility is associated with variations affecting the gene represented in this entry.

☒
TP53 is found in increased amounts in a wide variety of transformed cells. TP53 is frequently mutated or inactivated in about 60% of cancers. TP53 defects are found in Barrett metaplasia a condition in which the normally stratified squamous epithelium of the lower esophagus is replaced by a metaplastic columnar epithelium. The condition develops as a complication in approximately 10% of patients with chronic gastroesophageal reflux disease and predisposes to the development of esophageal adenocarcinoma.

The disease is caused by mutations affecting the gene represented in this entry.

The gene represented in this entry is involved in disease pathogenesis.

Showing 1 to 4 of 4 entries (filtered from 108 total entries)

Previous

1

Next

Show entries

A C57BL6J Fancg-KO Mouse Model Generated by CRISPRCas9 Partially Captures the Human Phenotype

Publication date: 2023-07-05

PubMed ID: 37446306

Tagged Genes: Fancg

☐

A Case of Aplastic Anemia and Colon Cancer With Underlying Spliceosome Mutation Is It an Incidental Finding or a Novel Association

Publication date: 2022-02-01

PubMed ID: 35371815

Tagged Genes: Mlh1

☐

A Clinical Conundrum with Diagnostic and Therapeutic Challenge a Tale of Two Disorders in One Case

Publication date: 2023-11-01

PubMed ID: 37526892

Tagged Genes: Dclre1c, Fanca

☐

A De Novo Frameshift Mutation in RPL5 with Classical Phenotype Abnormalities and Worsening Anemia Diagnosed in a Young Adult-A Case Report and Review of the Literature

Publication date: 2023-11-05

PubMed ID: 38004002

Tagged Genes: Rpl5

☐

A Single-Centre Experience of First-Line Romiplostim and Immunosuppressi... Therapy in Patients With Aplastic Anemia

Publication date: 2023-04-01

PubMed ID: 37206485

Tagged Genes: Tpo, Pc

☐

Showing 1 to 5 of 247 entries

Previous

1

2

3

4

5

...

50

Next

Reset

Apply

Close

This example workflow combines different ontologies to extract information from PubMed and adds Tags in an interactive view at the end.

Successfully Addressing Information Overload in Research Literature

In this section we discussed different approaches and options to undertake biomedical literature surveys using dictionaries and ontologies, tagging the abstracts according to those terms as well as displaying the results through an interactive visualization.

With this, we created a tool that can help to investigate for example hypotheses around adverse drug events with respect to ethnic backgrounds of a patient without going into different literature sources. Is this all that can be done using such approaches? Not at all! There are a lot more questions that could be tackled using such tools and extending the existing workflows with other sources, terms and ideas. This can ultimately lead to customizable and easily generalizable tools that can be applied to literature reviews in any domain.

Node & Topic Index

A

Adverse Event	42, 46, 81
API	36, 81
Autocomplete Widget.....	82, 85

B

Bag of Words Creator	47
Bootstrap Sampling.....	69

C

Cancer Treatment.....	42
Catch Errors	78
Classification	59, 67
Column Filter.....	28
CSV Writer.....	69

D

Data Access	36
Data Analysis	36, 42, 46, 59
Data App	3, 11, 15, 21, 27, 32
Database.....	46
Deep Learning	52, 59, 67
Dictionary Tagger	82
Disease Tracking	32
Document Grabber	75, 82
Drug Dosing	3, 11, 15, 21, 27
Drug Monitoring.....	3, 11, 15, 21, 27, 42

F

FAERS Data	42
Flow Variables	18

G

Generic JavaScript View	75
GET Request	32, 38

GroupBy	75
---------------	----

H

Hypotheses Testing	42
--------------------------	----

I

Interactive Dashboard	3, 11, 15, 21, 27, 32, 36
-----------------------------	---------------------------

J

JavaScript	73
JSON Path	38

K

Keras Batch Normalization Layer	63
Keras Convolutional 1D Layer.....	63
Keras Dense Layer.....	54, 63
Keras Flatten Layer	63
Keras Input Layer	54
Keras LSTM Layer	54
Keras Max Pooling 1D Layer.....	63
Keras Network	52, 59, 67
Keras Network Executor	54, 65, 70
Keras Network Learner	54, 65, 69
Keras Repeat Layer	54

L

Lag Column	54
Loop End.....	54
LSTM.....	52

M

Many to One	65, 70
Math Formula	18, 28

N

Network Viewer 78

P

Plotly 52
 Prediction 52
 PubMed Data 72, 73, 81
 Python 52, 59, 67

R

Round Double 28
 Rule Engine 28
 Rule Engine Variable 28

S

Scorer 65, 70
 Single Selection Widget 28
 SMOTE 69
 SPARQL Endpoint 85
 StanfordNLP NE Learner 75
 StanfordNLP NE Scorer 75

Statistics 62
 String Input 28
 SubGraph Extractor 78

T

Table View 22, 78
 Tagged Document Viewer 82
 Term Co-Occurrence Counter 78
 Text Processing 73, 81
 Tile View 7
 Triple File Reader 85
 Try 78

V

Vaccine 46
 VAERS 46
 Visualization 32, 36, 52, 72, 73

W

Web Service 32, 36, 81
 Widget 7, 18, 22, 28

Boosting Digital Health

Data-driven Applications for Digital Healthcare

To prevent diseases and improve quality of life, digital health applications use data and emerging technologies. This booklet collects examples of personalized treatments through data apps, the potential of using open, often unstructured, data sources, and artificial intelligence to make healthcare more data-driven.

Dr. Dayanjan (“Shanaka”) Wijesinghe is an associate professor at Virginia Commonwealth University, School of Pharmacy. His research focuses on the development of precision approaches to health with a focus on digital technologies. He teaches PhD and PharmD (Doctor of Pharmacy) students precision medicine, computer-aided drug discovery, digital health, and rapid prototyping of concepts related to healthcare using low-code approaches. Here, he writes about training future healthcare professionals in data science.

Ali Asghar Marvi is a Data Scientist in the Evangelism team at KNIME. He studied Computer Science as an undergraduate and developed a keen interest in the applications of Data Science. He recently obtained his master’s degree in Computer and Information Science from the University of Konstanz, Germany. At KNIME, Ali creates and writes about workflows that are meant to cover practical aspects of businesses.