

Open for Innovation

KNIME

Artificial Intelligence

Visual Programming for ~~Data Science~~

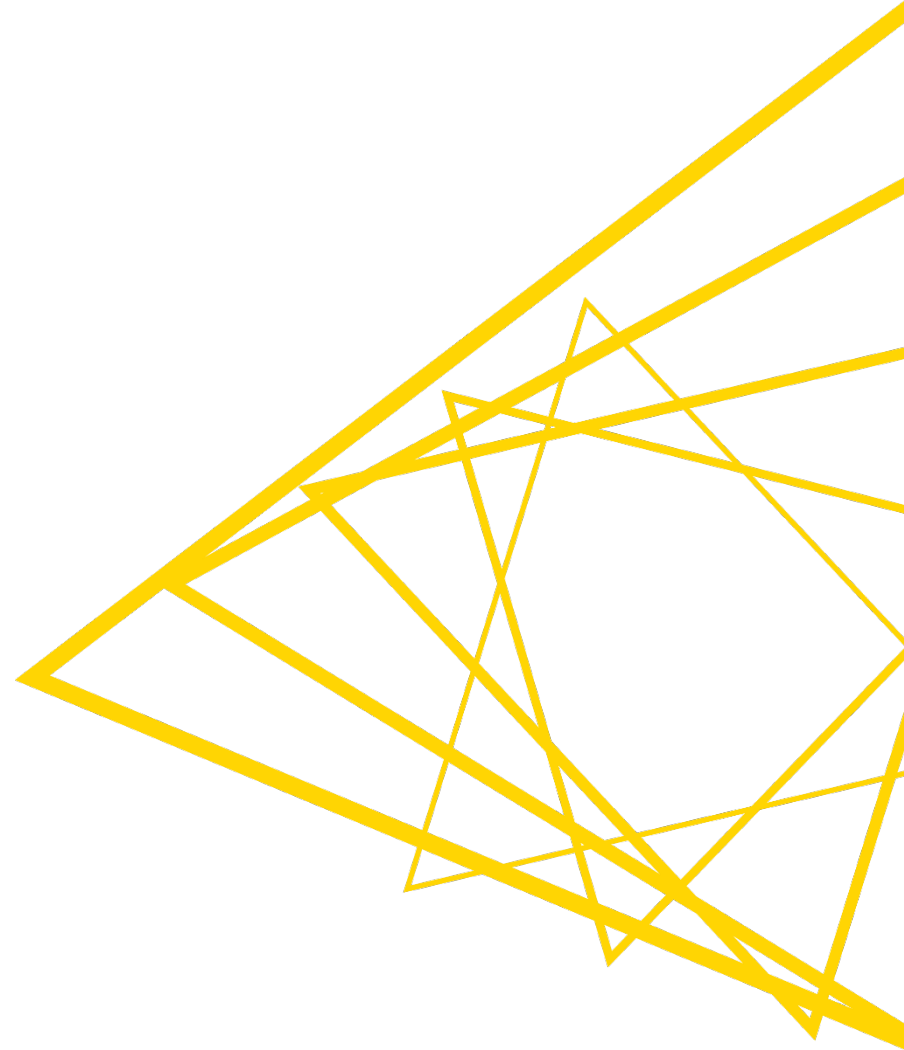
Working with Data at the right Level of Abstraction

Prof. Dr. Michael Berthold

ISACA Fokus Event, Nov 10, 2022



Data Science Personas



Data Science Researchers

Researchers focus on Data Science algorithms

- often have CS/NumMath/Stats background
- use Python, Javascript, R, SQL, C, Java, or XYZ
- tend to not worry about others.

```
#include <iostream>
#include <seqan/file.h>
#include <seqan/sequence.h>

using namespace seqan;

int computeLocalScore(String<char> subText, String<char> pattern)
{
    int localScore = 0;
    for (unsigned i = 0; i < length(pattern); ++i)
        if (subText[i] == pattern[i])
            ++localScore;

    return localScore;
}

String<int> computeScore(String<char> text, String<char> pattern)
{
    String<int> score;
    resize(score, length(text) - length(pattern) + 1, 0);

    for (unsigned i = 0; i < length(text) - length(pattern) + 1; ++i)
        score[i] = computeLocalScore(Infix(text, i, i + length(pattern)), pattern);

    return score;
}

int main()
{
    String<char> text = "This is an awesome tutorial to get to know SeqAn!";
    String<char> pattern = "tutorial";
    String<int> score = computeScore(text, pattern);

    for (unsigned i = 0; i < length(score); ++i)
```

```
library(foreach)
length_divisor<-6
iterations<-5000
predictions<-foreach(m=1:iterations,.combine=cbind)
%do% {
    training_positions <- sample(nrow(training),
    size=floor((nrow(training)/length_divisor)))
    train_pos<-1:nrow(training) %in% training_positions
    lm_fit<-lm(y~x1+x2+x3,data=training[train_pos,])
    predict(lm_fit,newdata=testing)
}
predictions<-rowMeans(predictions)
error<-sqrt((sum((testing$y-
predictions)^2))/nrow(testing))
```

```
class TICWritingConsumer : public MSDataWritingConsumer
{
public:
    TICWritingConsumer allows to change the
    filename (to "filename") using the processS
    functions.

    set TIC to zero
    filename : MSDataWritingConsumer(fi
    le = 0);

    step for spectra before they are writ
    dataWritingConsumer::SpectrumType & a)
    .size(); ++} { TIC += s[i].getIntensi
    ty; }

    RA processing
    (MSDataWritingConsumer::ChromatogramT
    ype & argv)

    if (argc < 2) return 1;
    // the path to the data should be given on the command line
    String tutorial_data_path(argv[1]);

    // Create the consumer, set output file name, transform
    TICWritingConsumer * consumer = new TICWritingConsumer("Tutori
    MzMLFile().transform(tutorial_data_path + "/data/Tutorial.FileIt
    em.txt", "TIC.txt");

    std::cout << "There are " << consumer->nr_spectra << " spectra
    std::cout << "The total ion current is " << consumer->TIC << std::endl;
    delete consumer;

    return 0;
} //end of main
```

```
*** contribution from Andrew Dalke ***
import sys
from rdkit import Chem
from rdkit.Chem import AllChem

# Download this from http://pypi.python.org/pypi/futures
from concurrent import futures

# Download this from http://pypi.python.org/pypi/progressbar
import progressbar

## On my machine, it takes 39 seconds with 1 worker and 10 seconds with 4.
## 29.055u 0.102s 0:28.68 101.6X 0+0k 0+3to 0pf+0w
max_workers=1

## With 4 threads it takes 11 seconds.
## 34.933u 0.188s 0:10.89 322.4X 0+0k 125+1to 0pf+0w
max_workers=4

# (The "user" time includes time spend in the children processes.
# The wall-clock time is 28.68 and 10.89 seconds, respectively.)

# This function is called in the subprocess.
# The parameters (molecule and number of conformers) are passed via a Python
def generateConformations(m, n):
    m = Chem.AddHs(m)
    ids = AllChem.EmbedMultipleConfs(m, numConfs=n)
    for id in ids:
        AllChem.UFFOptimizeMolecule(m, confId=id)
    # EmbedMultipleConfs returns a Boost-wrapped type which
    # cannot be pickled. Convert it to a Python List, which can.
    return m, list(ids)

smi_input_file, sdf_output_file = sys.argv[1:3]
n = int(sys.argv[3])
writer = Chem.SDWriter(sdf_output_file)
suppl = Chem.SmilesMolSupplier(smi_input_file, titleLine=False)

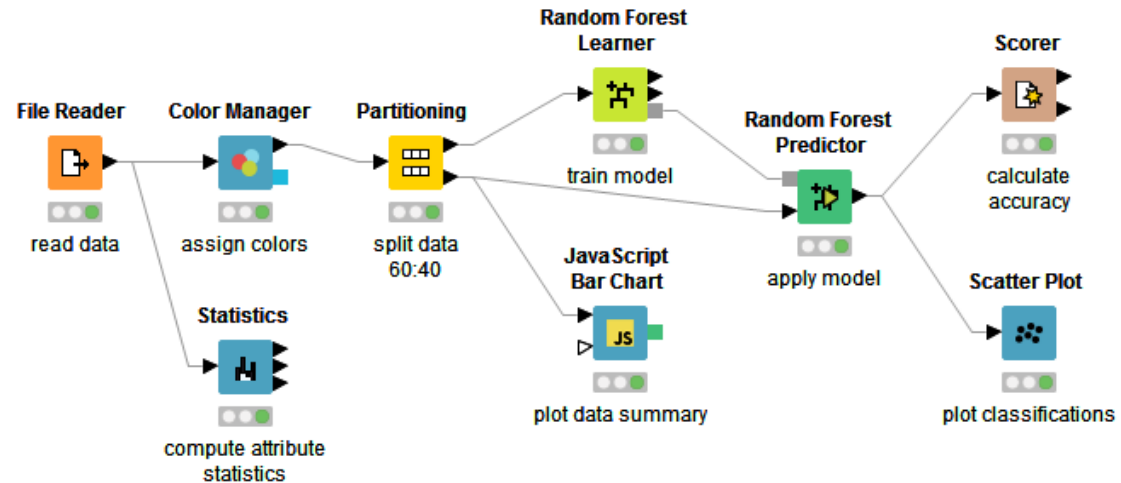
with futures.ProcessPoolExecutor(max_workers=max_workers) as executor:
    # Submit a set of asynchronous jobs
    jobs = []
    for mol in suppl:
        if mol:
            job = executor.submit(generateConformations, mol, n)
            jobs.append(job)

widgets = ["Generating conformations; ", progressbar.Percentage(), " ",
          progressbar.ETA(), " ", progressbar.Bar()]
pbar = progressbar.ProgressBar(widgets=widgets, maxval=len(jobs))
for job in pbar.futures.as_completed(jobs):
    mol, ids = job.result()
    for id in ids:
        writer.write(mol, confId=id)
writer.close()
```

Data Scientists

Data Scientists

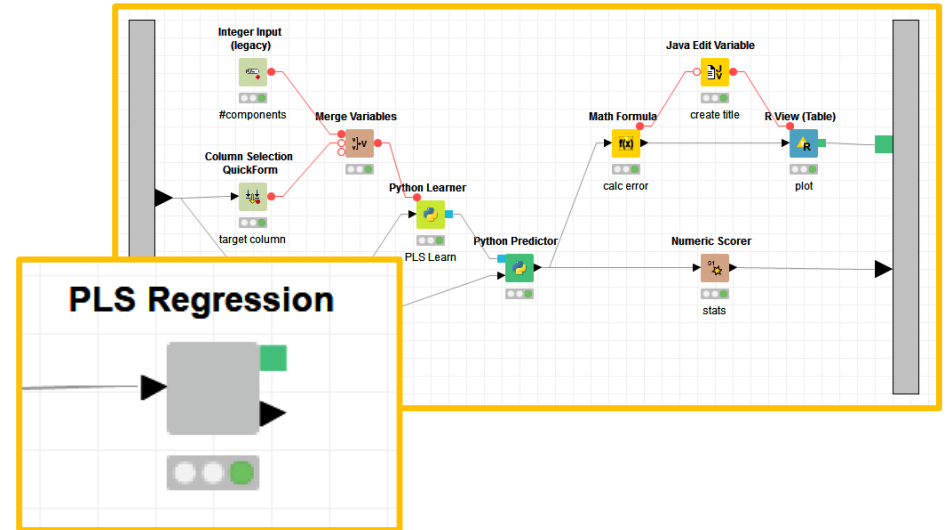
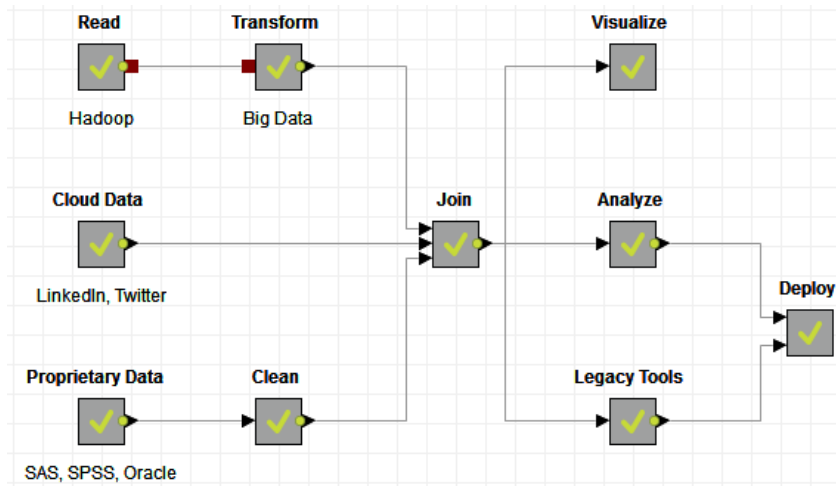
- have (also) a specific domain background
- often can program in Python, R, SQL, C, Java, or ...
- ...but really focus on the data flow
- want to use and try out other stuff



Casual Users (“Citizen Data Scientists”)

The occasional data person

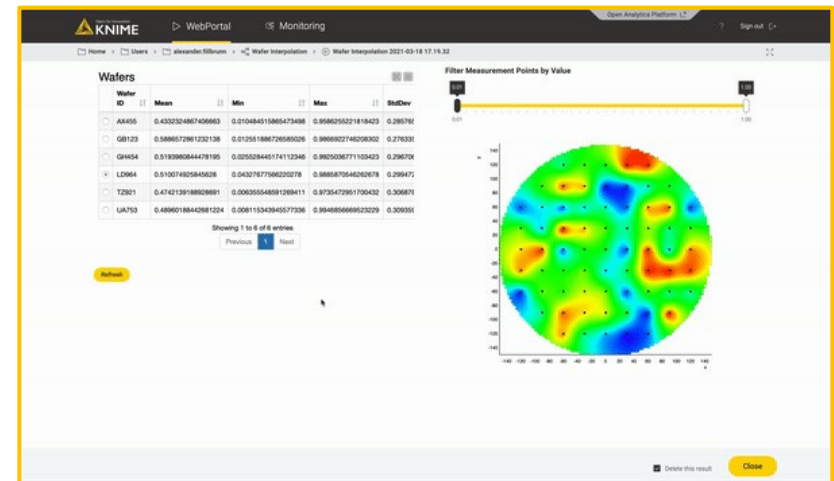
- wants to start from something existing that solves a similar problem
- often comes from an Excel, BI, or other data world
- needs solid documentation and proper abstraction



Data Science Consumers

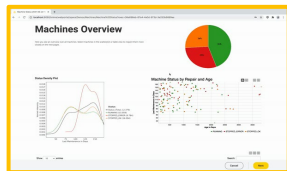
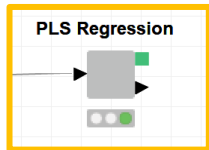
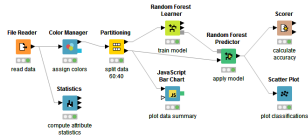
Data Science Consumers use

- their favorite frontend
(= Data Science needs to be embedded via an API)
- or
- Interactive Analytics (= Data Science Applications)



Data Science Personas

```
library(foreach)
length_of_rows<-5
iterations<-5000
predictions<-foreach(m=1:iterations, combine=cbind) %do% {
  training_positions <- sample(nrow(training),
  size=length(nrow(training)/length_of_rows))
  train_pos<-1:nrow(training) %in% training_positions
  lm_fit<-lm(y~x1+x2+x3,data=training[train_pos,])
  predict(lm_fit,newdata=testing)
}
predictions<-rowMeans(predictions)
error<-sqrt((sum((testing$y-
predictions)^2))/nrow(testing))
```



Coders:
Programs & Scripts

Data Scientists:
Visual Programming

Citizen Data Scientists:
Components & Blueprints

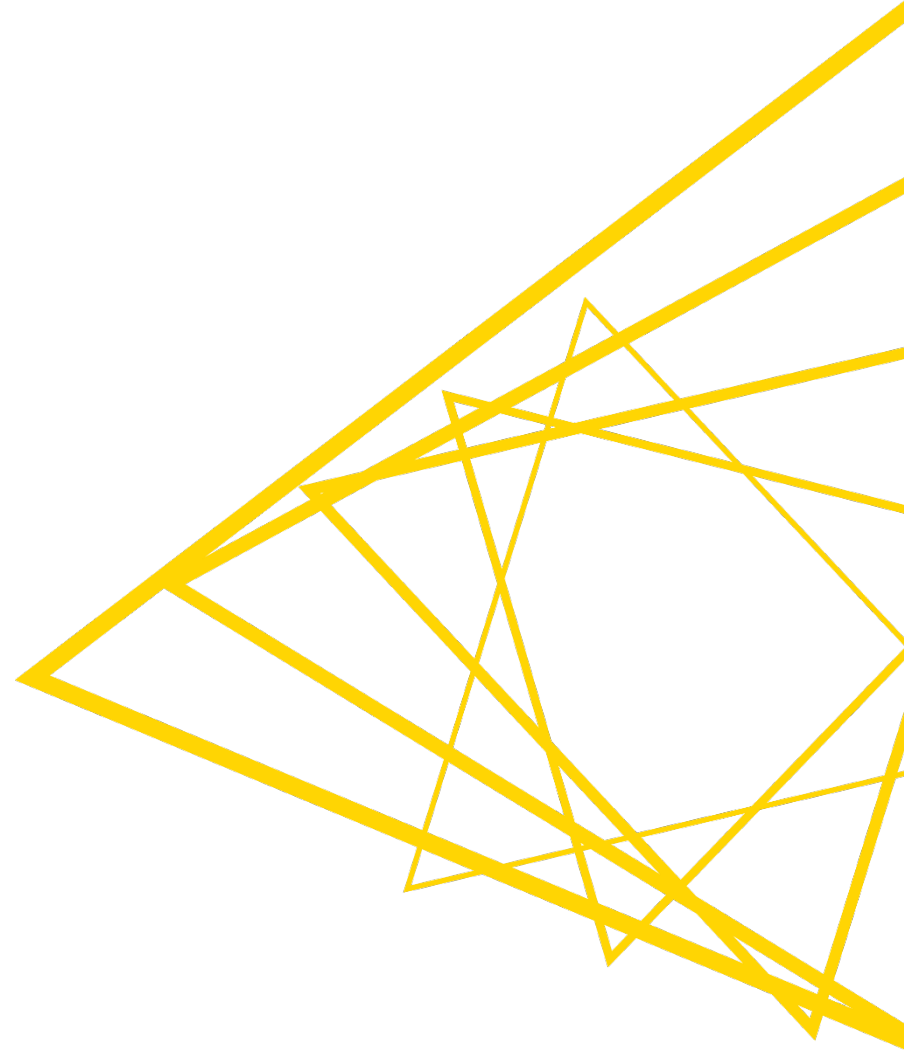
Data Science Consumers:
Applications & Services

Data Science
Algorithm Inventors

Data Science
Creators

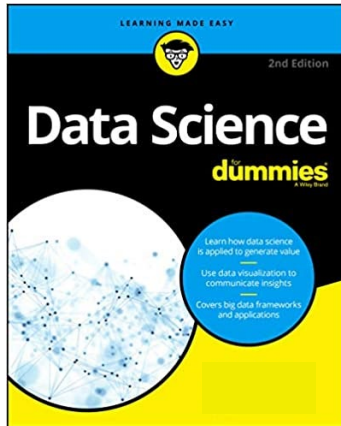
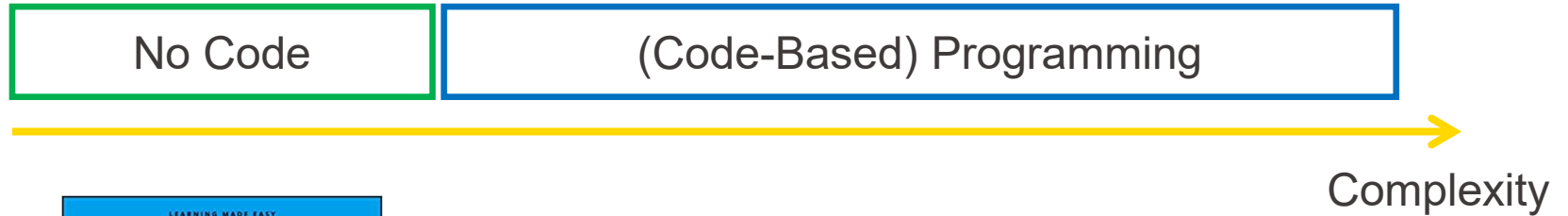
Data Science
Consumers

**No Code, Low Code,
Visual Programming?**



No Code

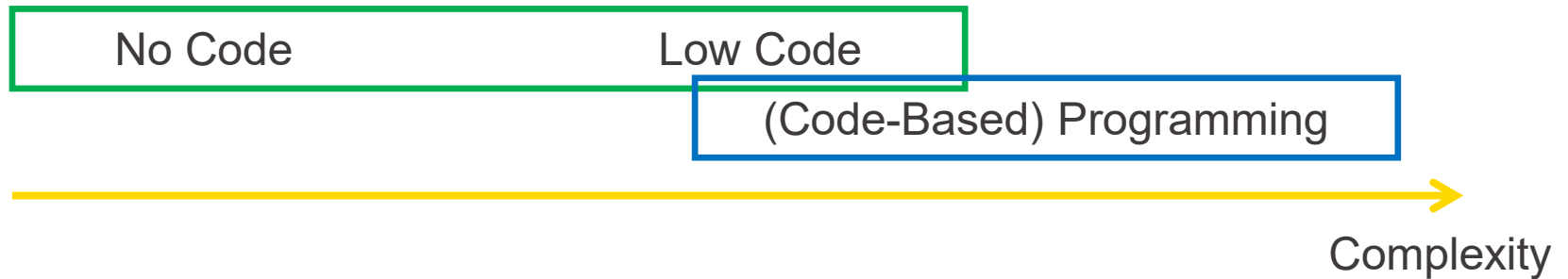
No Code for Simple Problems:



Visual Environment allows quick creation of solutions for standard problems.

Low Code

Low Code for Standard Problems:



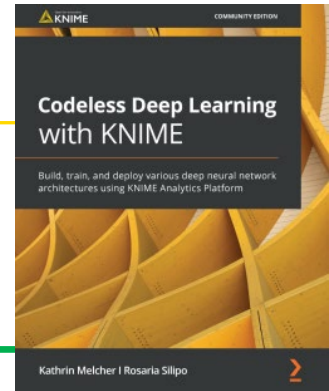
Visual Environment allows 80% creation of solutions.

...for the rest (& details): reach out to code.

(often: Visual UI on top of one programming language.)

Visual Programming

Visual Programming for all types of Problems:



Visual Programming

(Code-Based) Programming

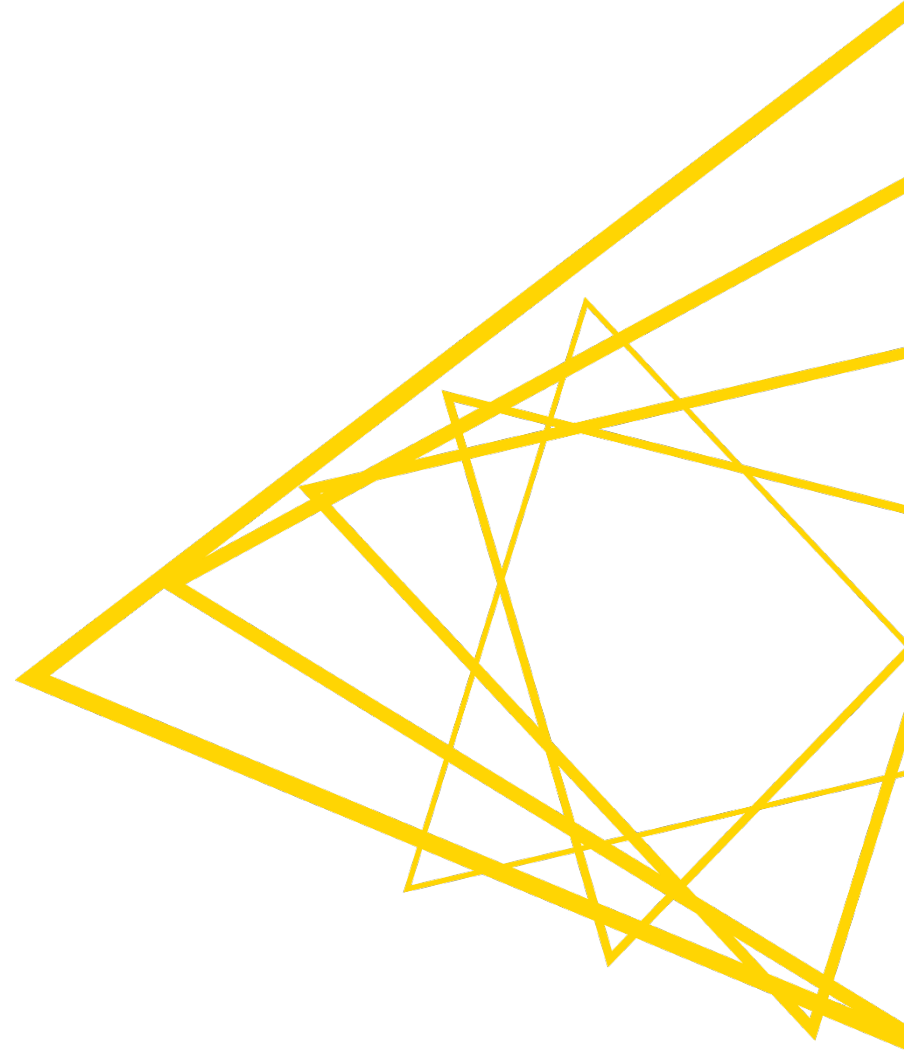
Complexity

Visual Environment allows complete creation of solutions.
(from simple to complex)

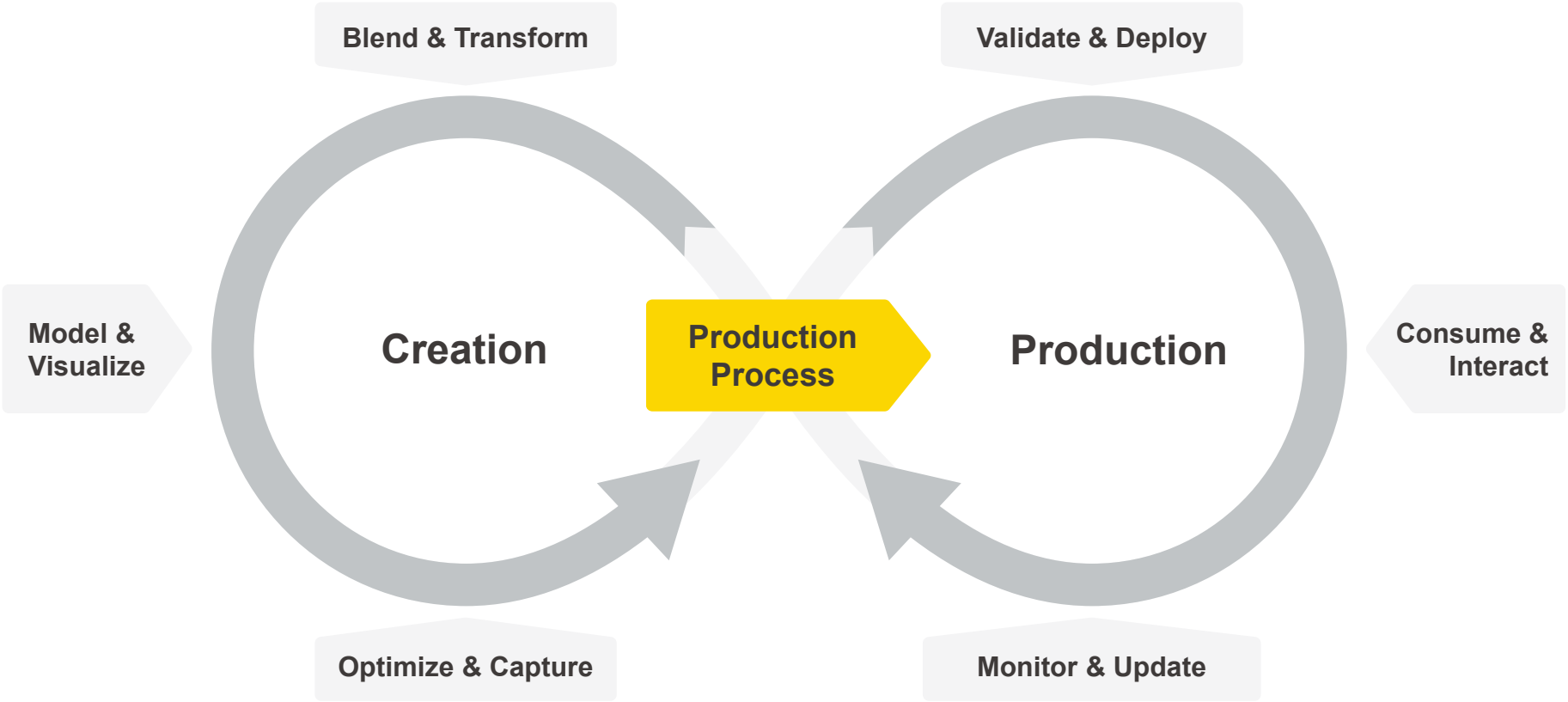
If desired: you can also include code (of various languages!).

(Visual UI models what's running under the hood.)

Visual Programming for Data Science



The Data Science Life Cycle



Data Science Activities



Data Wrangling
(Virtual Warehouses)



Descriptive Analytics
(Automating Repetition)



Diagnostic Analytics
(Creating Insights)

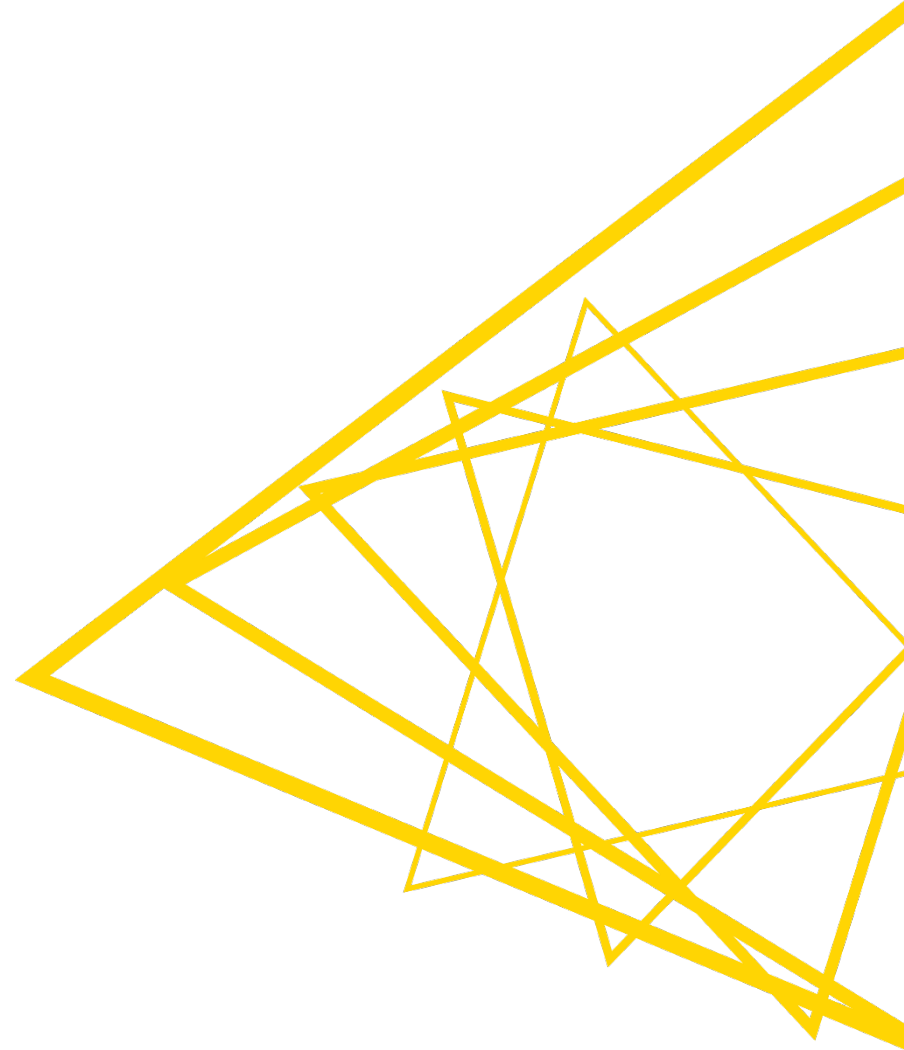


Advanced Analytics (AI/ML...)
(Predictive, Prescriptive)



Productionize
(Deploy, Manage, Govern)

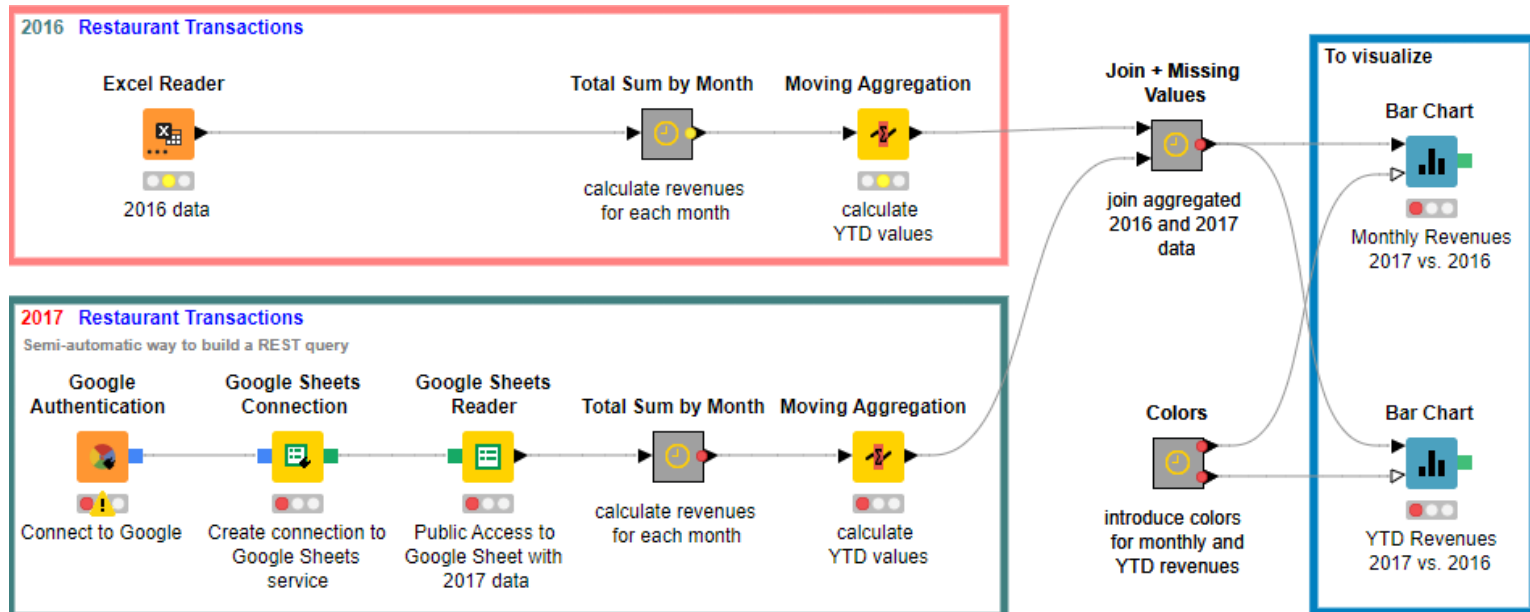
No Code for Analytics



No Code for Analytics

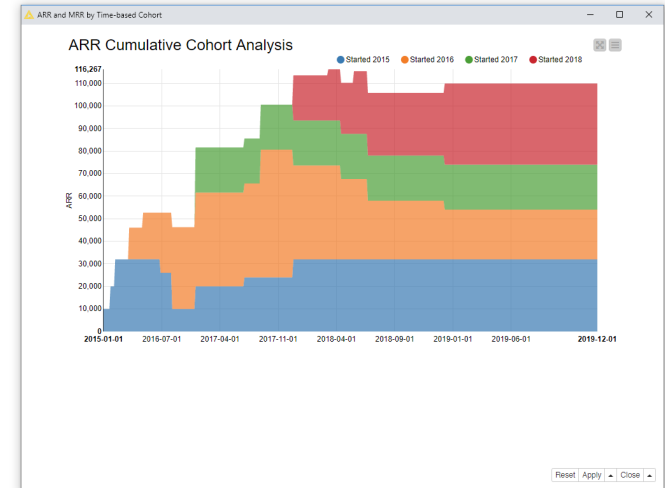
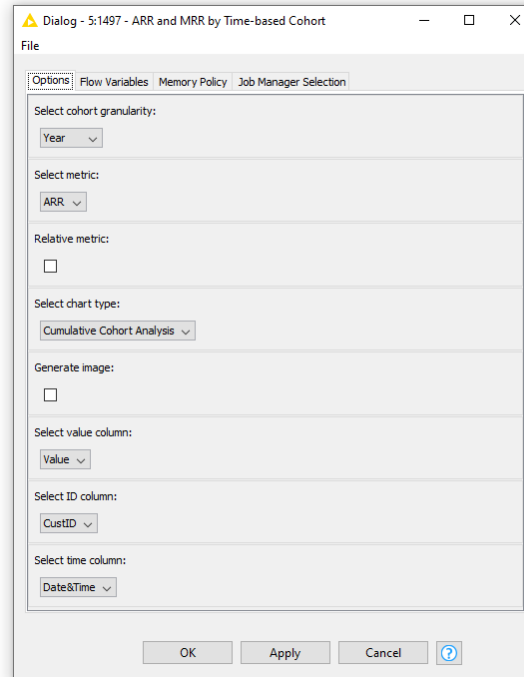
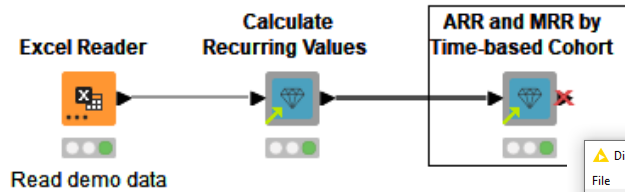
Benefits:

- (Reliable) Automation & Reproducibility
- Documentation



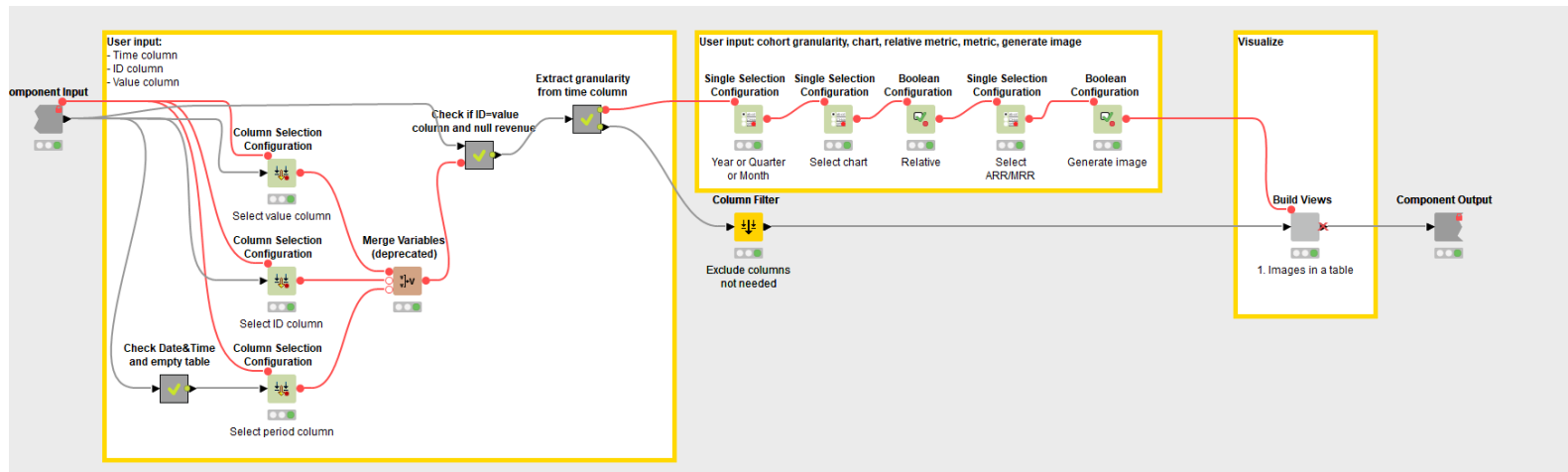
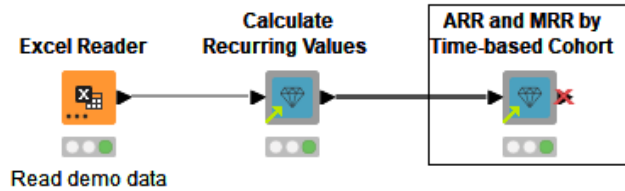
Creating Data Insights: Abstraction helps

ARR Cohort Analysis for Excel Users

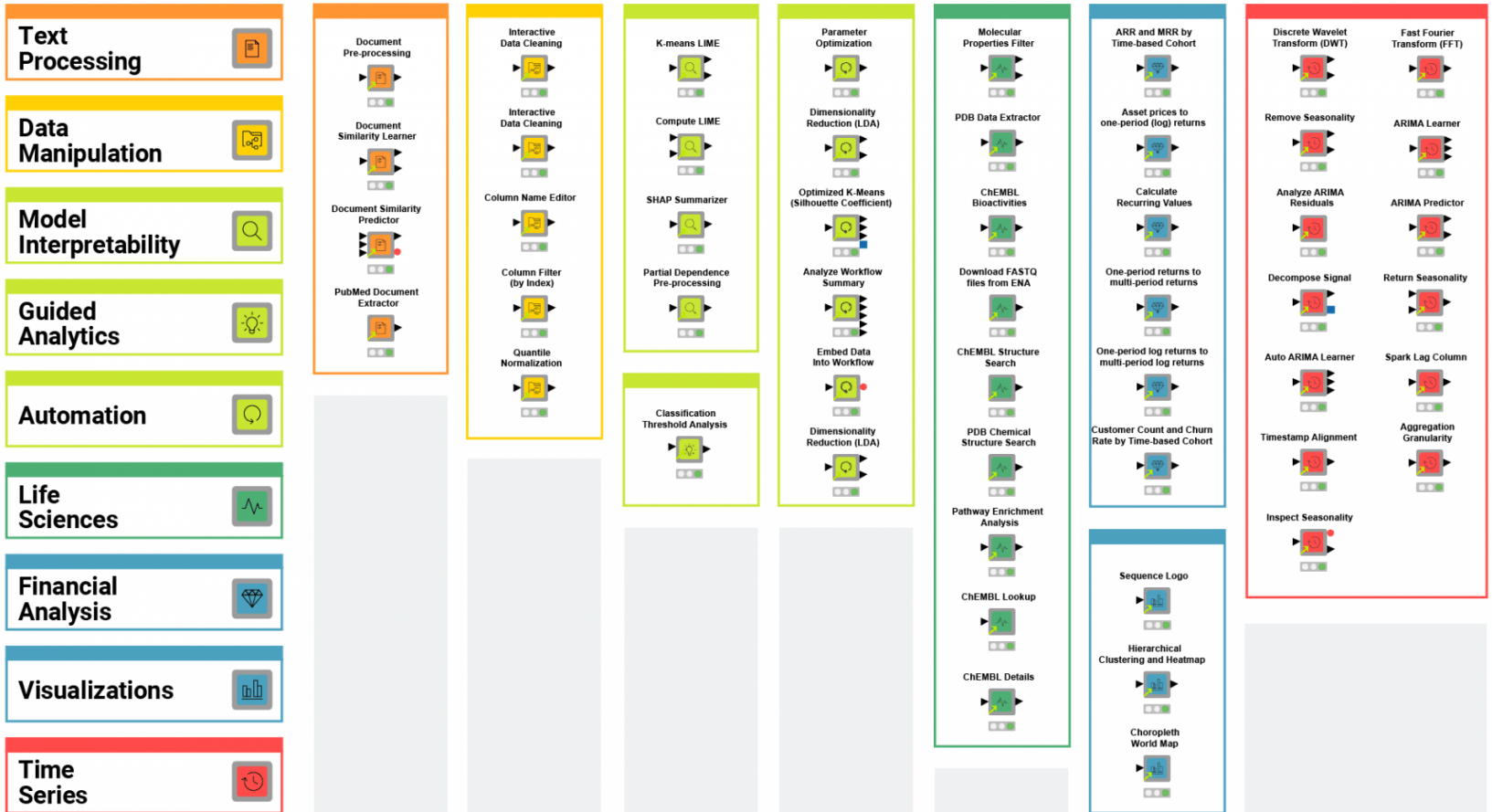


Creating Data Insights: Abstraction helps

ARR Cohort Analysis for Excel Users – No Code Encapsulated:



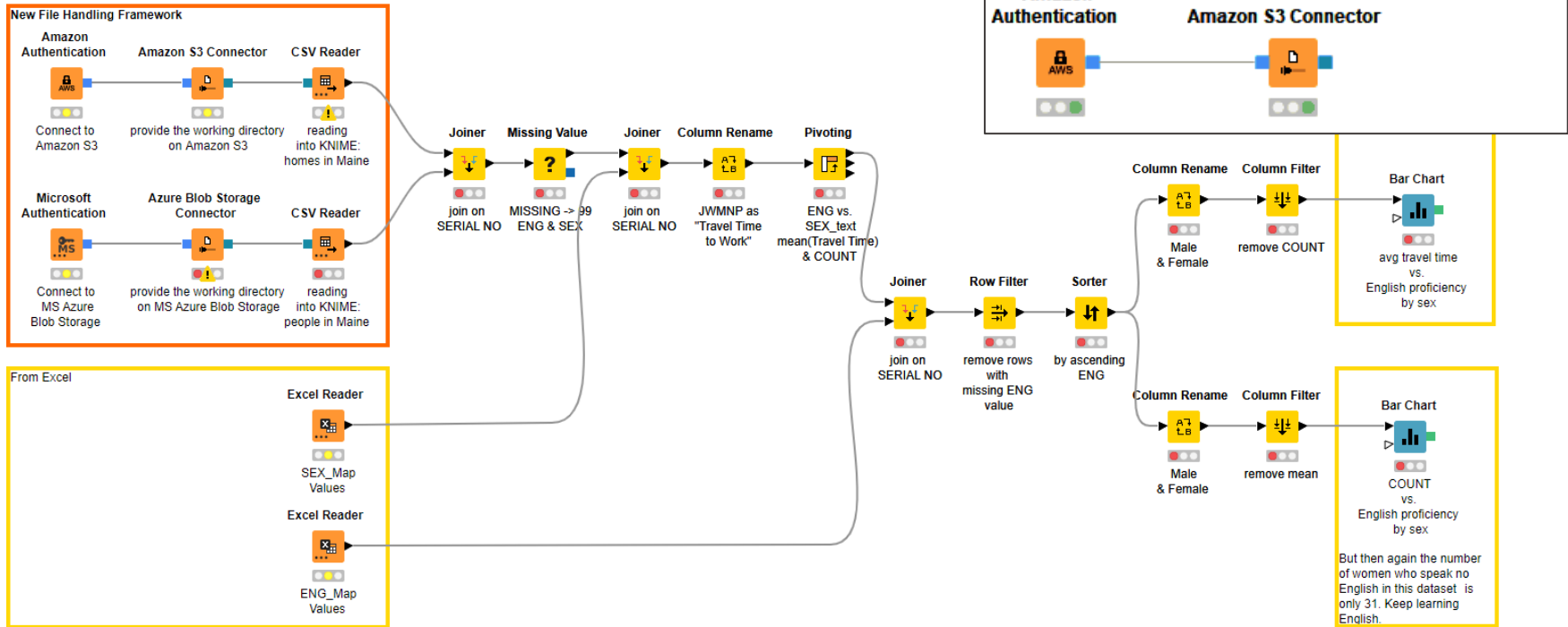
Abstraction in KNIME: Components



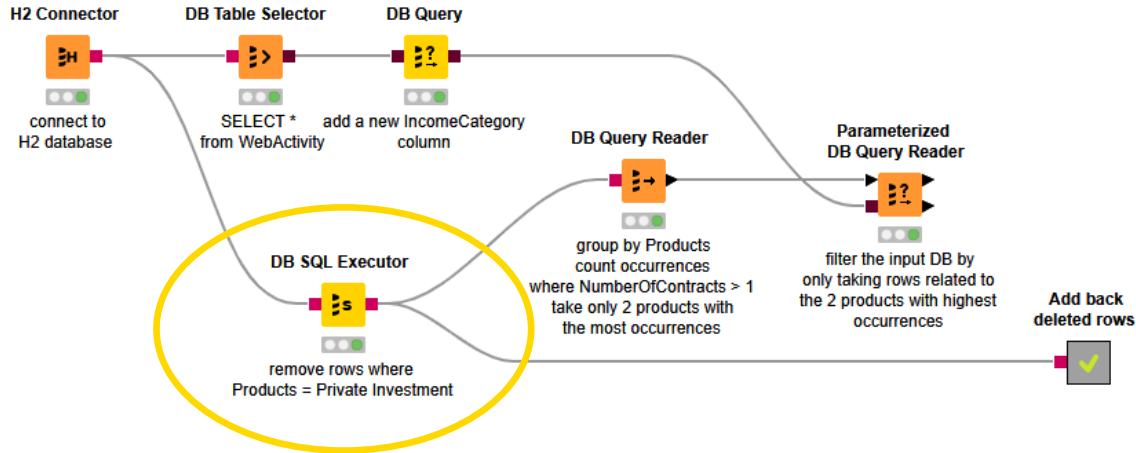
Low Code for Data Wrangling



Flexible Data Wrangling



Flexible Data Wrangling

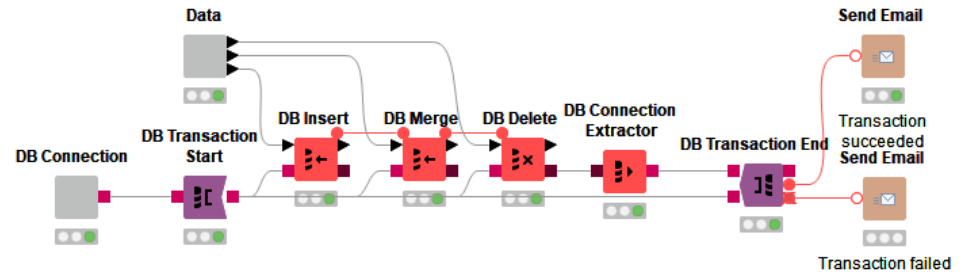


Connectors

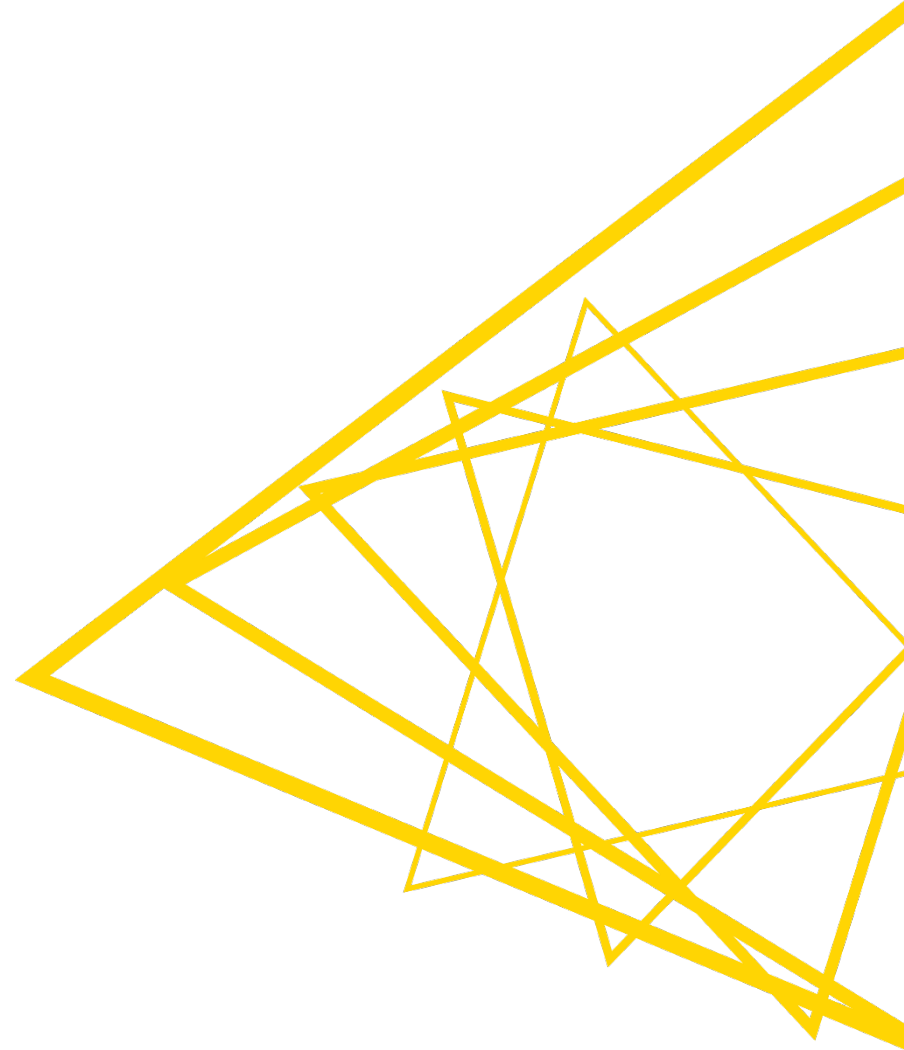
- Amazon S3 Connector
- Azure Blob Storage Connector
- Azure Data Lake Storage Gen2 Connector
- Databricks File System Connector
- FTP Connector
- Google Cloud Storage Connector
- Google Drive Connector
- HDFS Connector
- HDFS Connector (KNOX)
- HTTP(S) Connector
- KNIME Server Connector
- KNIME Workflow Data Area Connector
- Local File System Connector
- Microsoft Authentication
- SMB Connector
- SSH Connector
- SharePoint Online Connector

DB

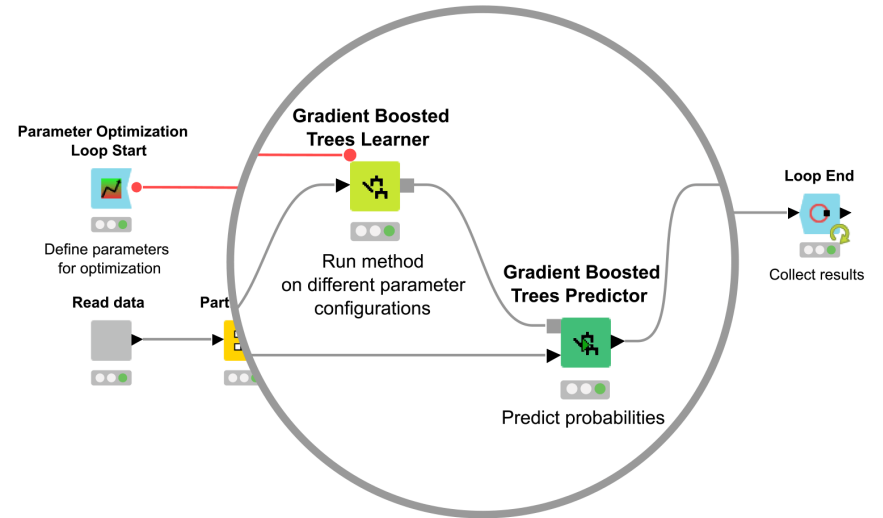
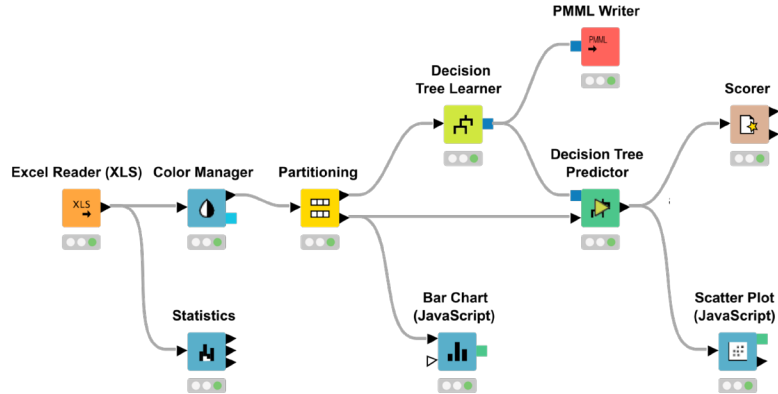
- Connection
- Amazon Athena Connector
- Amazon Redshift Connector
- DB Connection Closer
- DB Connection Extractor
- DB Connector
- H2 Connector
- Hive Connector
- Impala Connector
- Microsoft Access Connector
- Microsoft SQL Server Connector
- MySQL Connector
- Oracle Connector
- PostgreSQL Connector
- SQLite Connector
- Snowflake Connector
- Vertica Connector



Low Code for Data Science



No Code for Data Science



No Code for Interactive Visualizations

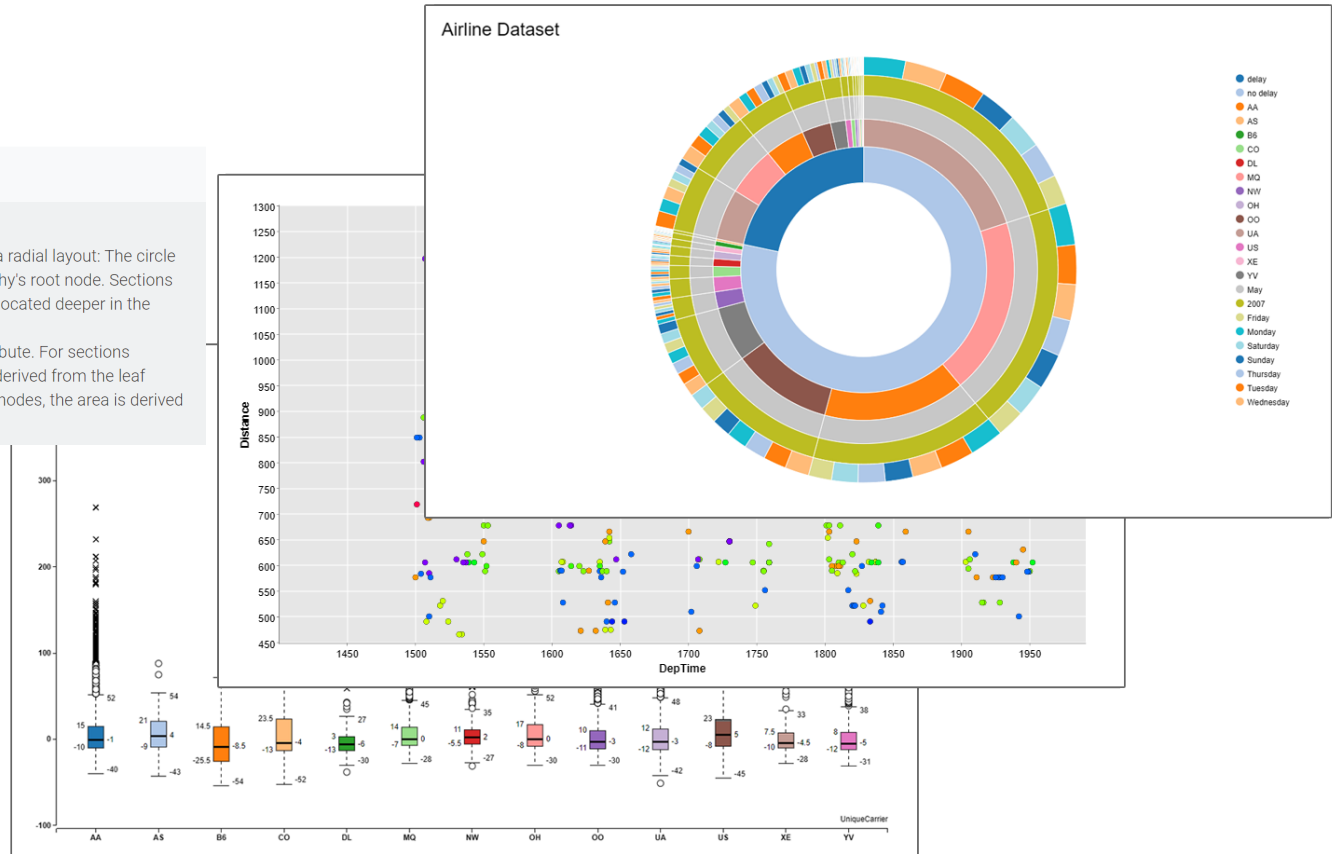
KNIME Hub > Nodes > Sunburst Chart

Node / Visualizer

Sunburst Chart

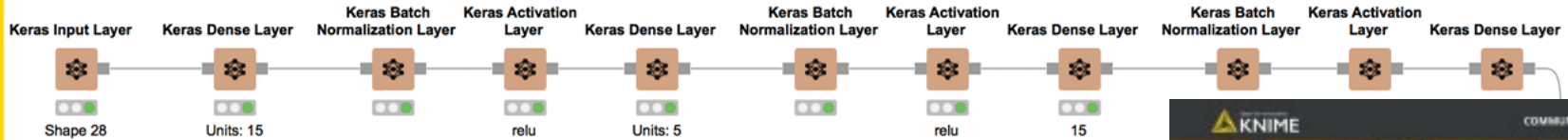


This chart displays hierarchical data in a radial layout: The circle in the chart center represents the hierarchy's root node. Sections further outside represent nodes that are located deeper in the hierarchy. Each leaf node has an attached value attribute. For sections corresponding to leaf nodes, the area is derived from the leaf node's value attribute. For sections not corresponding to leaf nodes, the area is derived from the accumulated value of all descending leaf nodes.

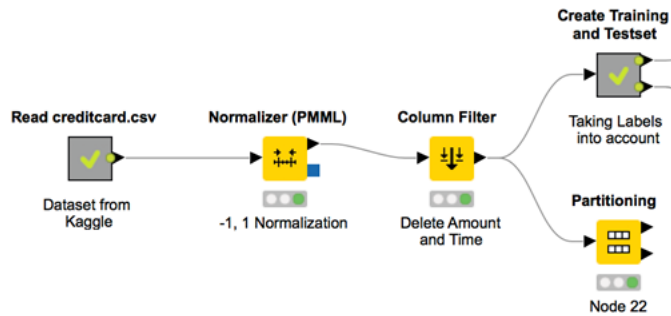


Codeless Deep Learning

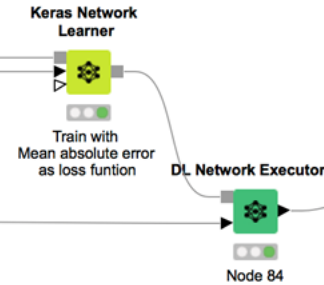
Define Network



Preprocessing



Training and Predicting



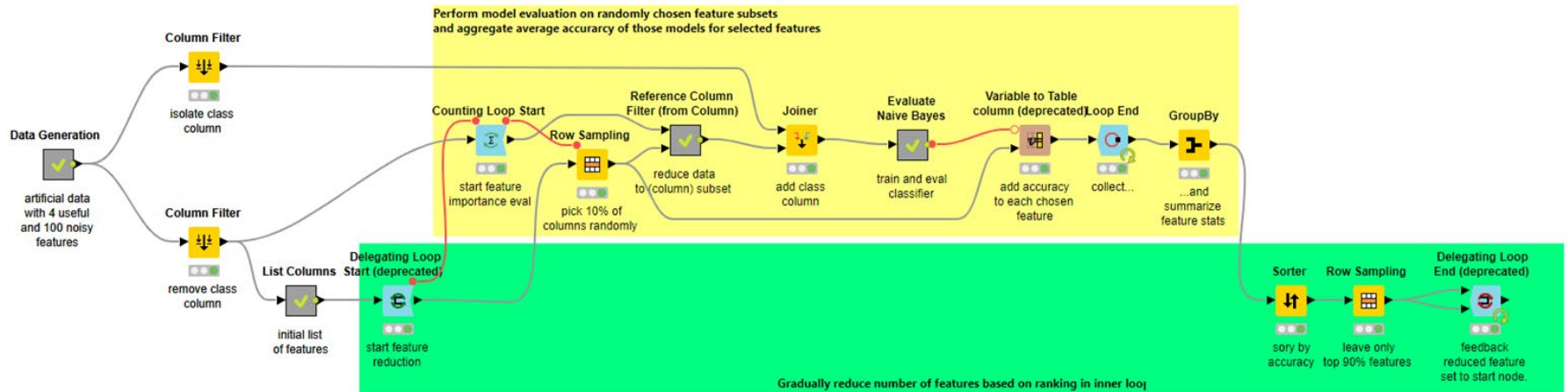
Codeless Deep Learning with KNIME

Build, train, and deploy various deep neural network architectures using KNIME Analytics Platform

Kathrin Melcher | Rosaria Sillipo

Data Science *is* complex: Visual Programming

Visual Data Science Algorithms: Recursive Feature Elimination

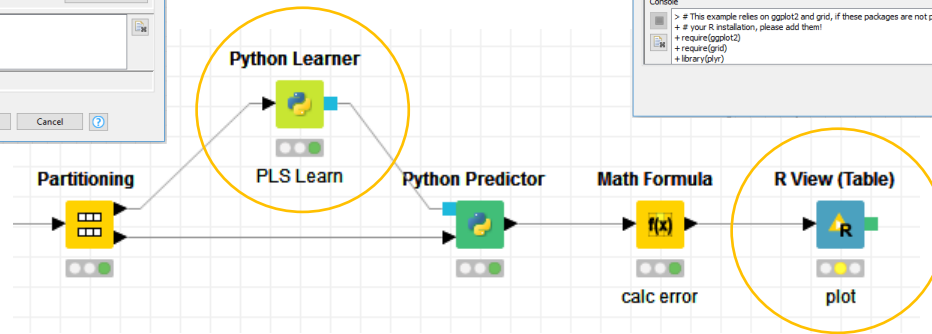
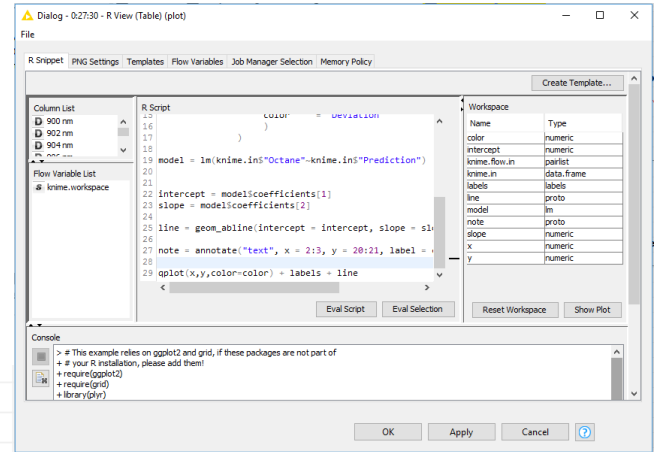
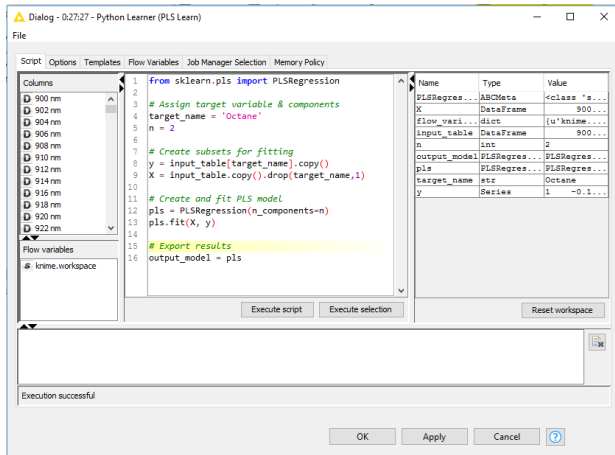


“Computer programming is an art, because it applies accumulated knowledge to the world, because it requires skill and ingenuity, and especially because it produces objects of beauty.”

Donald Knuth 1974

Low Code for Data Science

Leveraging Code in Visual Programming:



...works with (and from...) Jupyter Notebooks as well.

Data Science *is* complex: More Topics

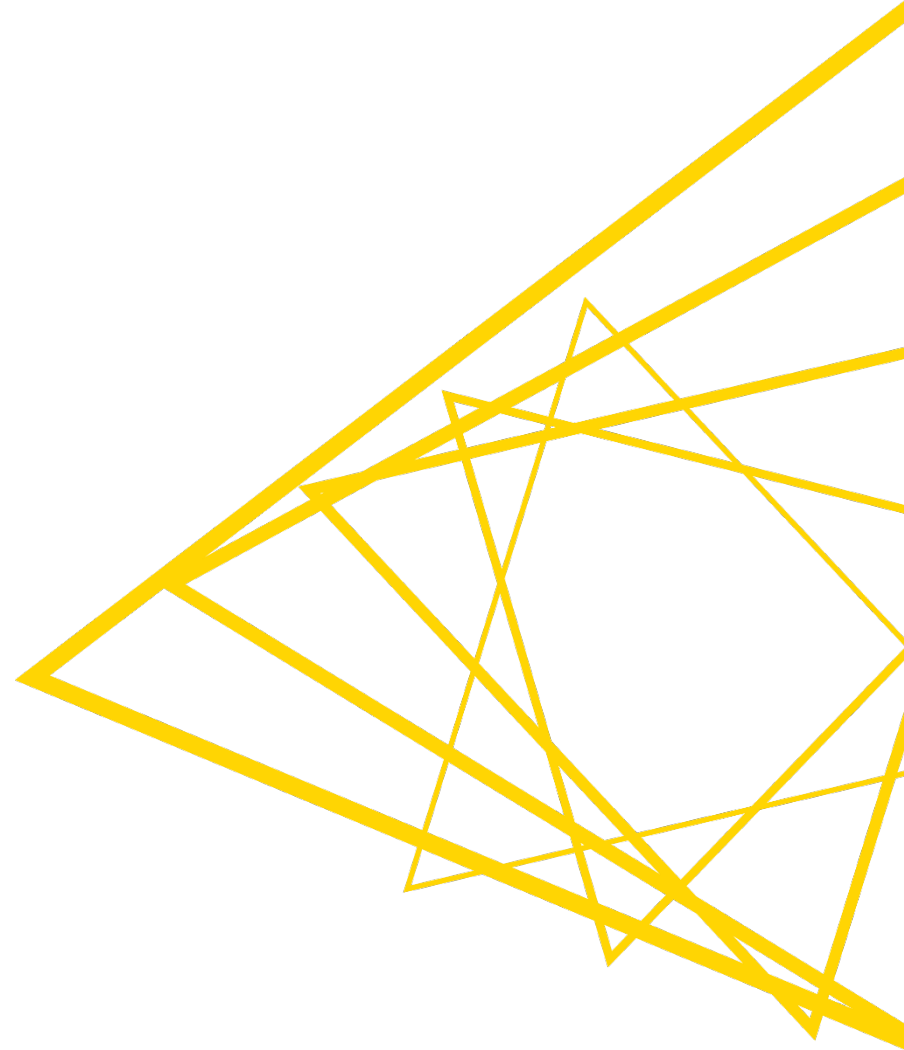
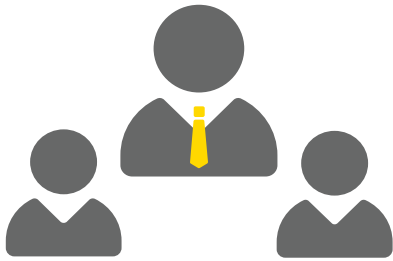
Data Science “Stuff”:

- Data Blending:
 - Feature Selection
 - Feature Engineering
 - Anonymization
 - ...
- Modeling
 - Model Selection
 - Parameter Optimization
 - Ensemble Creation
 - Active Learning
 - Transfer Learning
 - ...

Management Requirements:

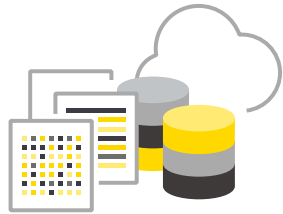
- Compliance
 - Best Practices
 - Required Standards
 - Bias Detection and Removal
 - Data Privacy / GDPR
 - Explainable “AI”
 - ...
- Governance
 - Integration / Dependencies
 - Traceable / Lineage
 - Documented
 - Reproducibility
 - ...

Data Science Teams



The Data Science Life Cycle

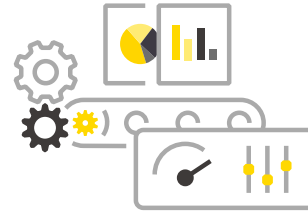
Blend & Transform



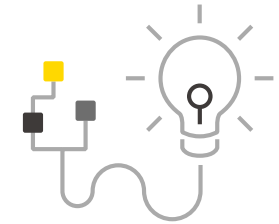
Model & Visualize



Deploy & Manage



Consume & Interact



Techniques:

- Joins
- Pivot
- Normalization
- PCA
- ...

- Clustering
- Regression
- Deep Learning
- Visualizations
- ...

- Model Monitoring
- Updating
- Validation
- ...

- Active Learning
- Interactive Applications
- Services
- ...

Technologies and Languages:

- SQL
- Data Lakes (AWS, Azure, Google...)
- R, Python
- Java, C, JavaScript, ...
- Go
- Kubernetes
- Docker
- Grafana
- ...
- Plotly, R Shiny
- JavaScript
- Vue.js
- REST
- ...

Data Science Teams shouldn't really have to worry about all of this...

Democratizing Data Science

What's the Essence of the Job:

- Create (complex) Data Science processes, often together with other Experts

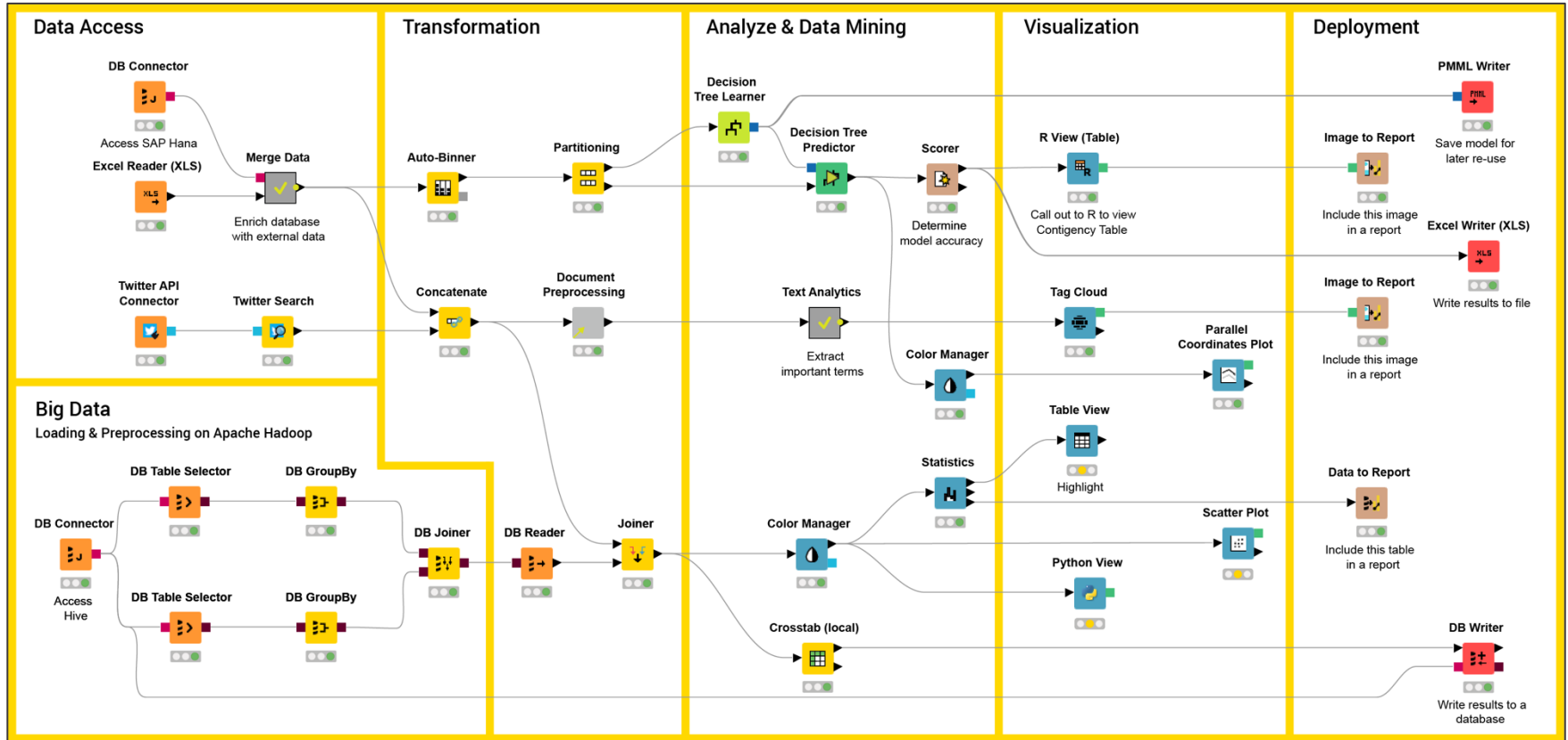
What's usually not part of the Job:

- Inventing, Writing, Optimizing new Algorithms

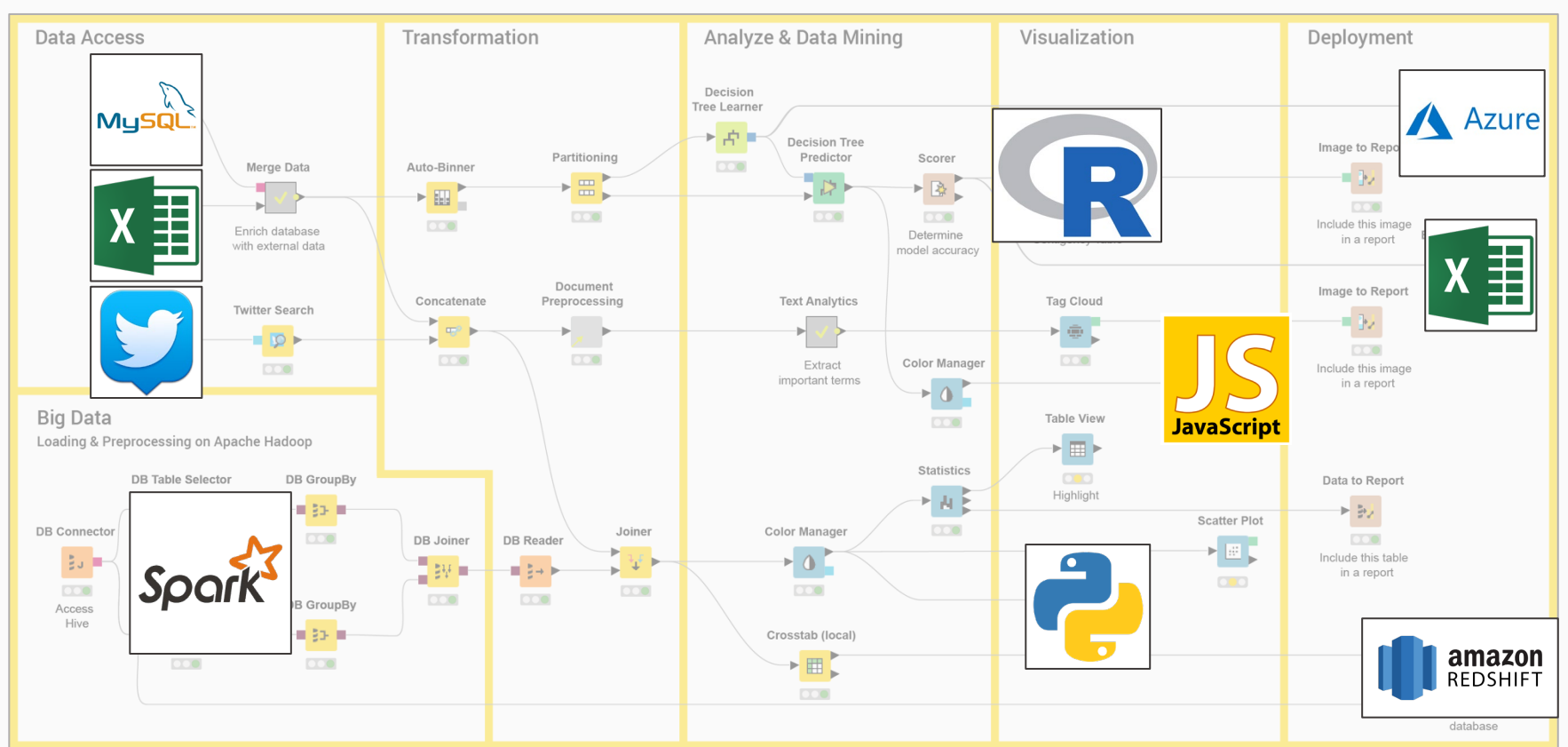
What shouldn't be part of the Job:

- Caring about the details of the underlying technology
- Worrying about interfaces of different tools
- Worrying about library versions and (backwards) compatibility
- Manually doing repetitive tasks

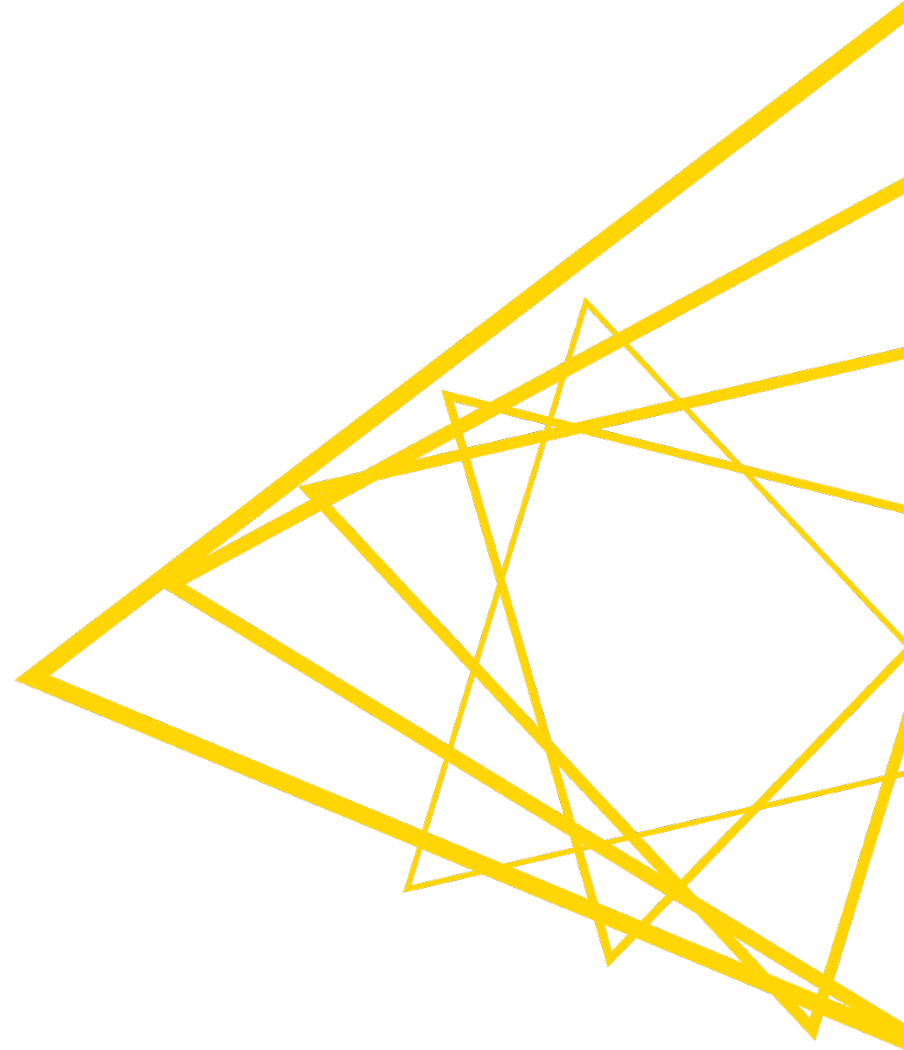
Visual Programming for Data Science



Technologies & Languages under the Hood



Low Code for Deployment



No Code Deployment of Data Apps

Cities

City	Latitude	Longitude
<input checked="" type="radio"/> Konstanz	47.6779	9.1732
<input type="radio"/> Zurich	47.3769	8.5417
<input type="radio"/> Berlin	52.5200	13.4050
<input type="radio"/> Austin	30.2672	97.7431
<input type="radio"/> New York	40.7128	74.0060
<input type="radio"/> Budapest	47.4979	19.0402
<input type="radio"/> Bologna	44.4949	11.3426

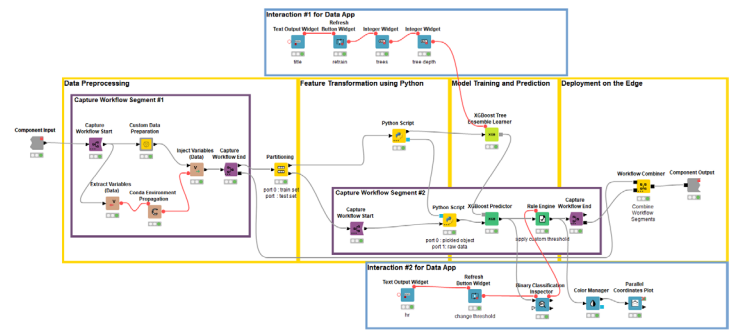
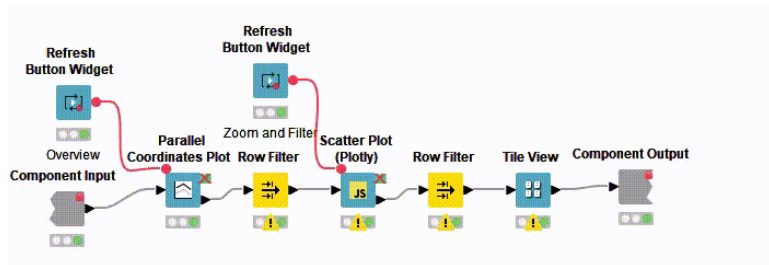
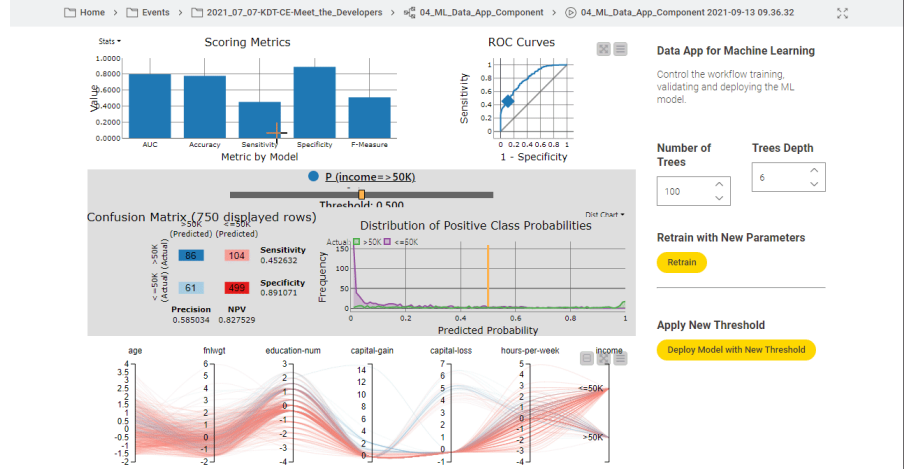
Showing 1 to 7 of 7 entries

Select City

People

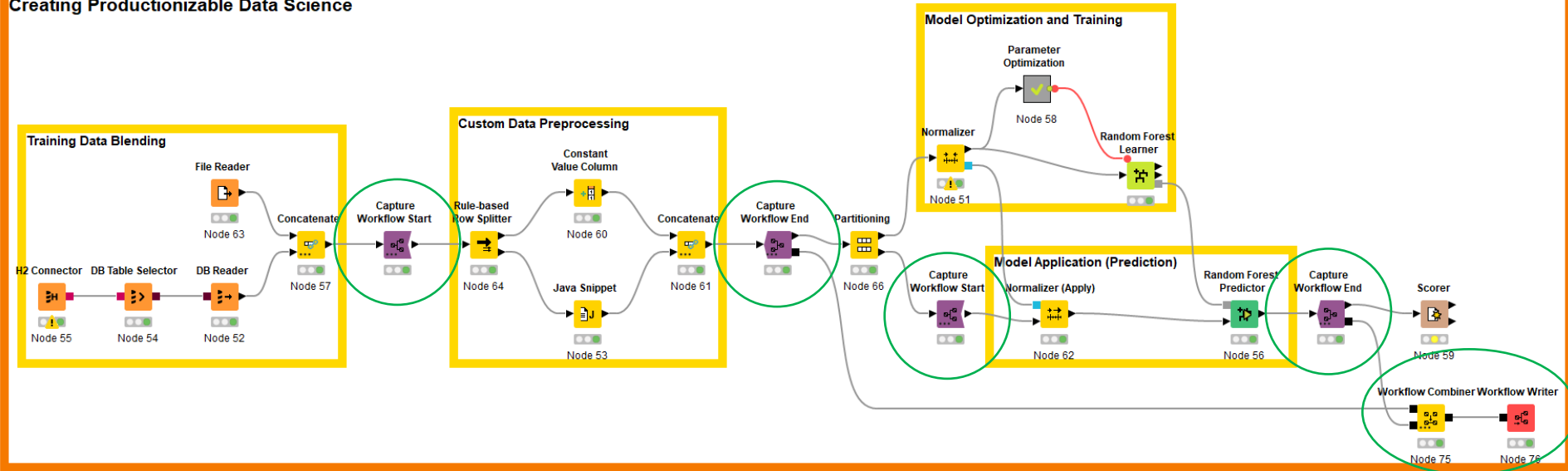
No data available in table

Showing 0 to 0 of 0 entries



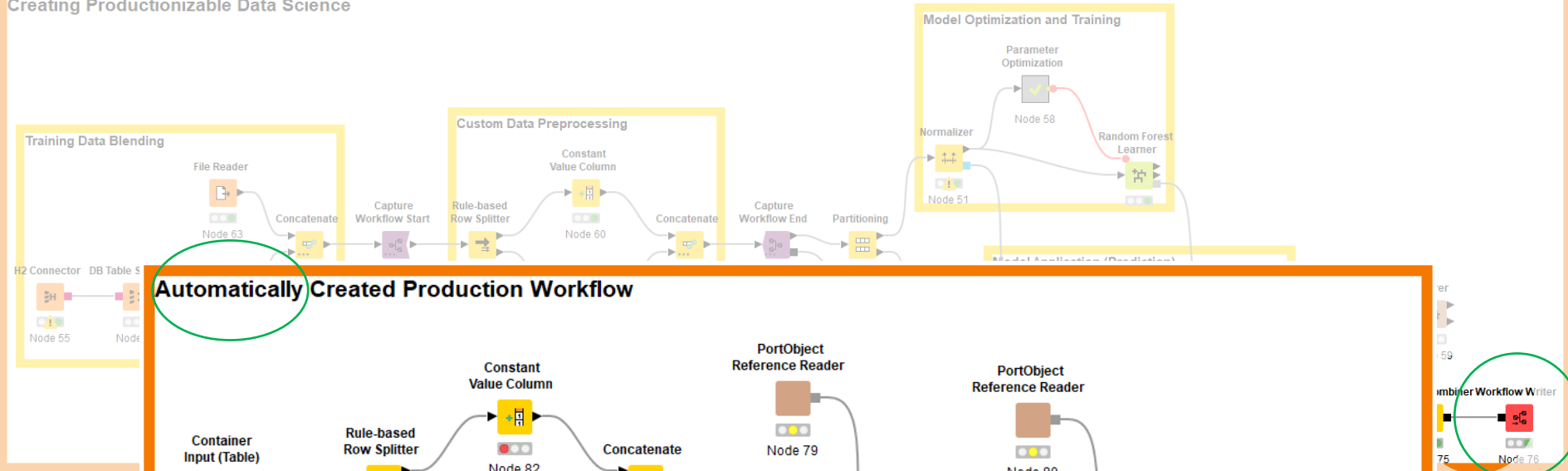
Deploying Data Science as Service

Creating Productionizable Data Science

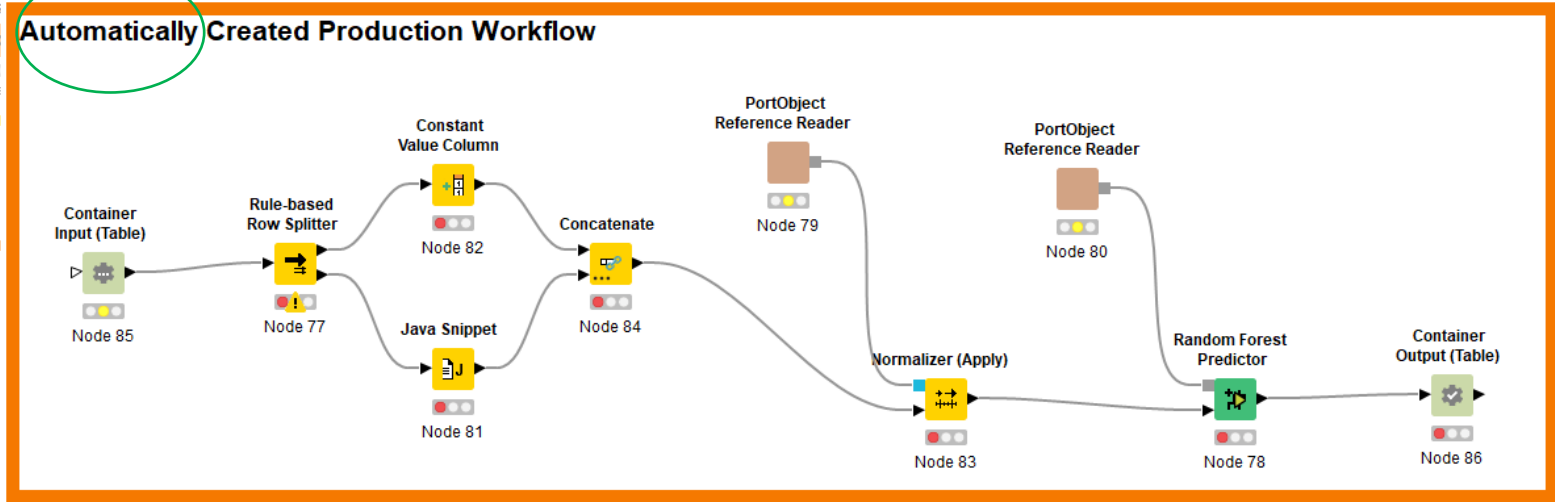


Deploying Data Science as Service

Creating Productionizable Data Science



Automatically Created Production Workflow



Visual Programming for all Aspects of Data Science



Data Wrangling
(Virtual Warehouses)



Descriptive Analytics
(Automating Repetition)



Diagnostic Analytics
(Creating Insights)



Advanced Analytics (AI/ML...)
(Predictive, Prescriptive)



Productionize
(Deploy, Manage, Govern)

Visual Programming for Data Science makes Sense!

- Data Science is a Team Sport:
 - requires a broad spectra of skills – often done in collaboration
 - requires understanding and access to the inner wheels of an algorithm (but not to the algorithm itself)
 - No single language or tool has all the answers – it's about tool blending as well!
- No-Code, Low-Code, Visual Programming (if done right):
 - automates repetitive tasks
 - provides the appropriate level of abstraction to allow focus on modeling a data process
 - allows access to all relevant nuts and bolts
 - allows to abstract / encapsulate sophistication for others to safely reuse
 - removes tool interaction/interface complexity
 - provides transparency & governance
 - (and it's not just a UI slapped on top of a language)
 - ...allows everybody to gradually increase their Data Science Proficiency (in the same Environment).

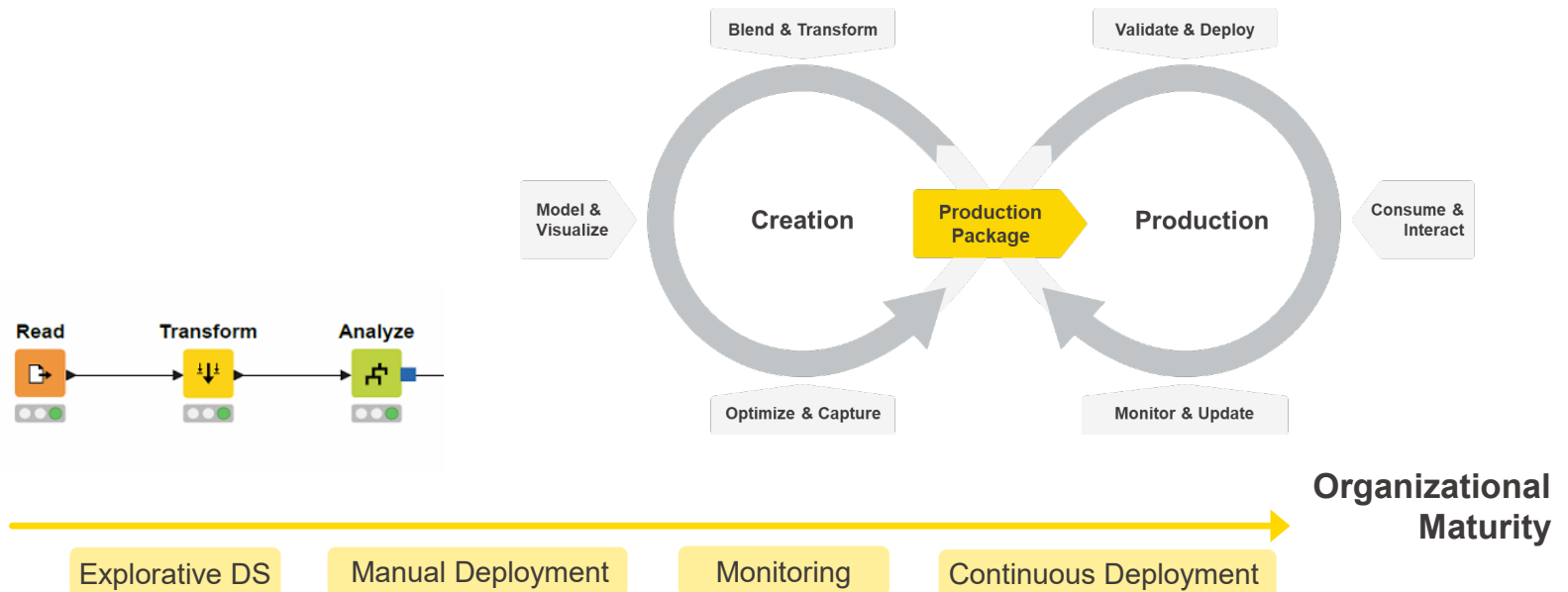
Data Science Upskilling

Individual Sophistication

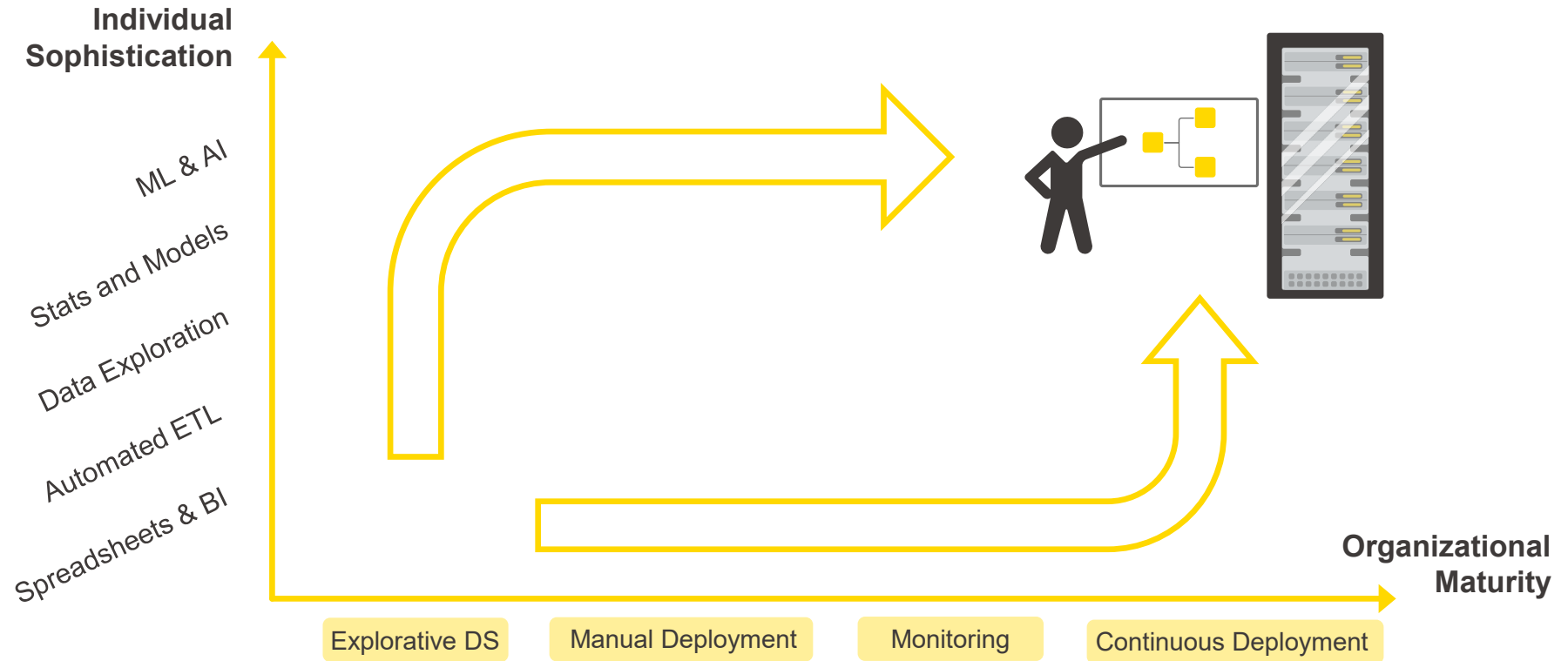
ML & AI
Stats and Models
Data Exploration
Automated ETL
Spreadsheets & BI



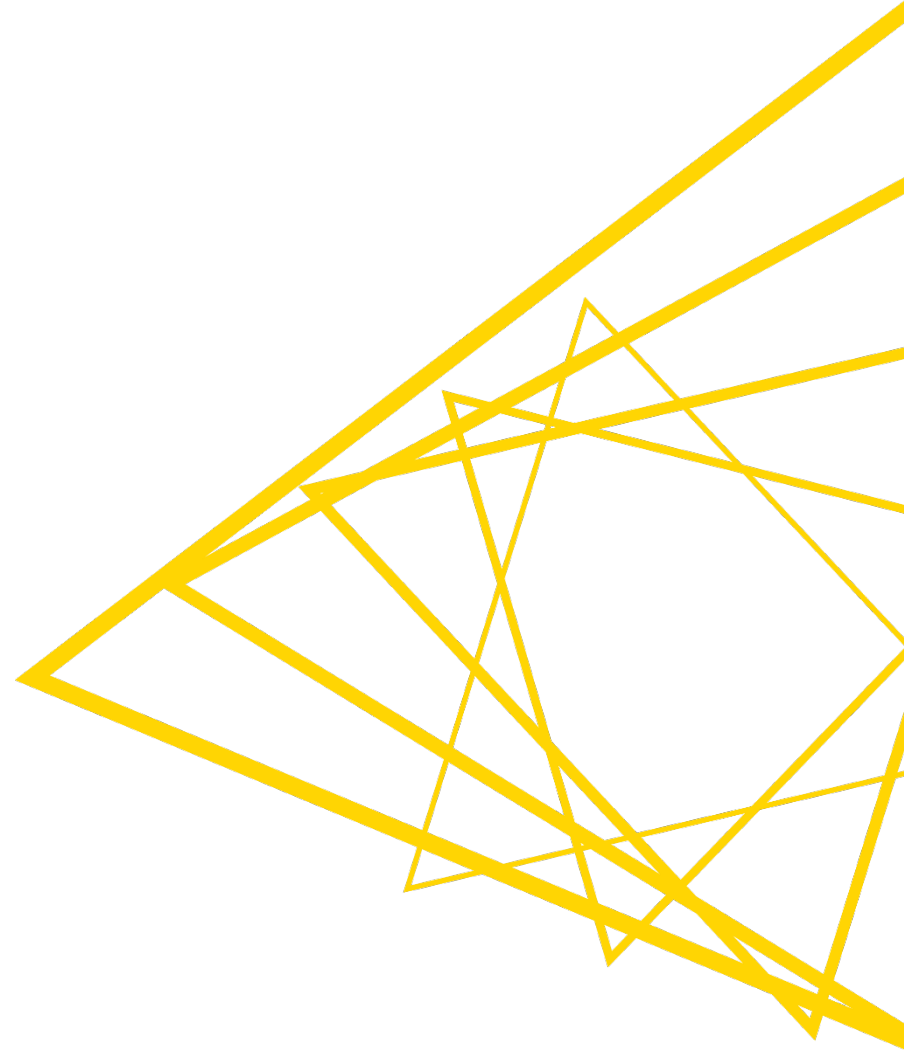
Organizational Upskilling



Upskilling Individuals and Organizations



Want to get started?



Getting Started?



Beginners corner

[L1-DS] KNIME Analytics Platform for Data Scientists: Basics

- Content 38 Modules
- Difficulty Basic
- Course Length 8 hours
- Rating ★★★★★ 1079 Reviews
- Reviewer/Instructor Maarit Widmann

Start

48 results

Filter: Beginner | Column Filtering | Data access | Data visualization | Tool workflow | Getting started | <-

All | Nodes | Components | Workflows | Extensions

Read Data from Microsoft Azure Cloud
This workflow demonstrates how to read data files (Text, Excel, KNIME Table and CSV) from the Microsoft Azure Cloud.

Read Data from Amazon S3
This workflow demonstrates how to read data files (Text, Excel, KNIME Table and CSV) from the Amazon S3.

Transform Data using GroupBy and Joiner nodes
In this workflow, a number of ETL operations are performed on the sales2009-2011 csv dataset. Besides showing what CTL, features A...

Welcome to the **KNIME Hub**

First Steps to Building Workflows

Example-based learning as a beginner can help you acquire new skills faster. Support from the online community share their workflows here for you to download and inspect. We collected a dedicated set of workflows by KNIME in the Beginners space below. These workflows are accompanied by a beginner cheat sheet and take you through your first steps to read, explore, transform, and deploy data in KNIME. Explore the links below.

Explore relevant resources

- The Beginners space on the KNIME Hub
- A cheat sheet on building workflows for beginners
- Getting Started Guide

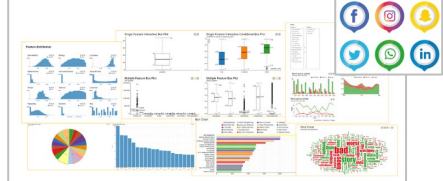
<https://www.knime.com/learning>

KNIME for Beginners

Data Exploration in #66daysofdata with KNIME

A roadmap to deepen your knowledge of data exploration techniques within the #66daysofdata challenge

Co-author: *Rosaria Silipo*



Have you heard about the #66daysofdata challenge?

KNIME Certified

Basic Proficiency
in KNIME Analytics Platform

LT | DT | IQ

KNIME

Do you need free notifications when people reply to your posts?

KNIME Analytics Platform

KNIME Analytics Platform Release Notes

KNIME Analytics Platform category

Just KNIME It!

Prove your KNIME knowledge and practice your workflow building skills by solving our weekly challenges.



KNIME

What is Data Aggregation?

PLAY ALL

Basic KNIME

31 videos • 108,603 views • Updated today

Here you can find videos about basic issue KNIME, like hitting, file reading, and so on.

You Tube

Low Code for Advanced

Home | About | Editor's Picks | Getting Started | Theory

Getting Started

In Low Code for Advanced Data Science

Medium

KNIME® BEGINNER'S LUCK

KNIME Analytics Platform for Beginners

Authors: Roberto Pellegrini and Thomas Hain