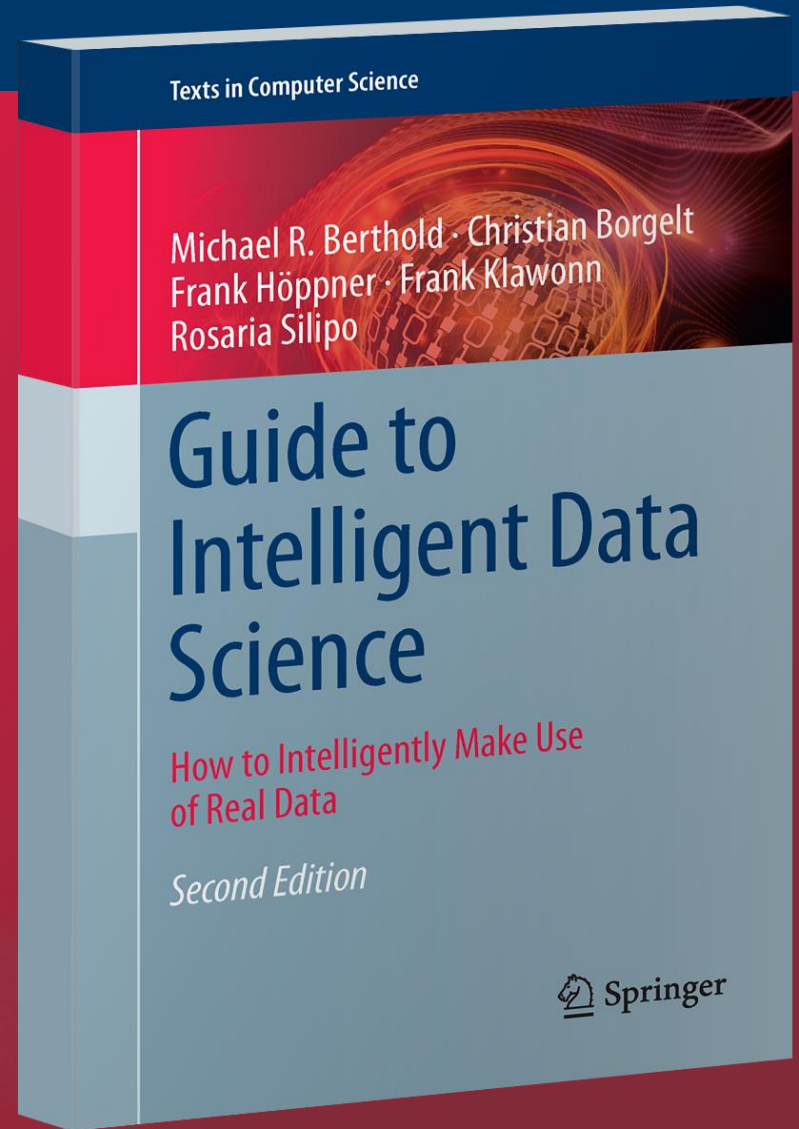


# Deployment



*„Data Scientist is just a sexed up word for Statistician“  
-Nate Silver*

How do we move the models to production?

*\*This lesson refers to chapter 10 of the GIDS book*

## Content of this Lesson

- Deployment
- Model Deployment
- Model Management
- Practical Example

# Deployment

# The Data Science Process

## — SEMMA

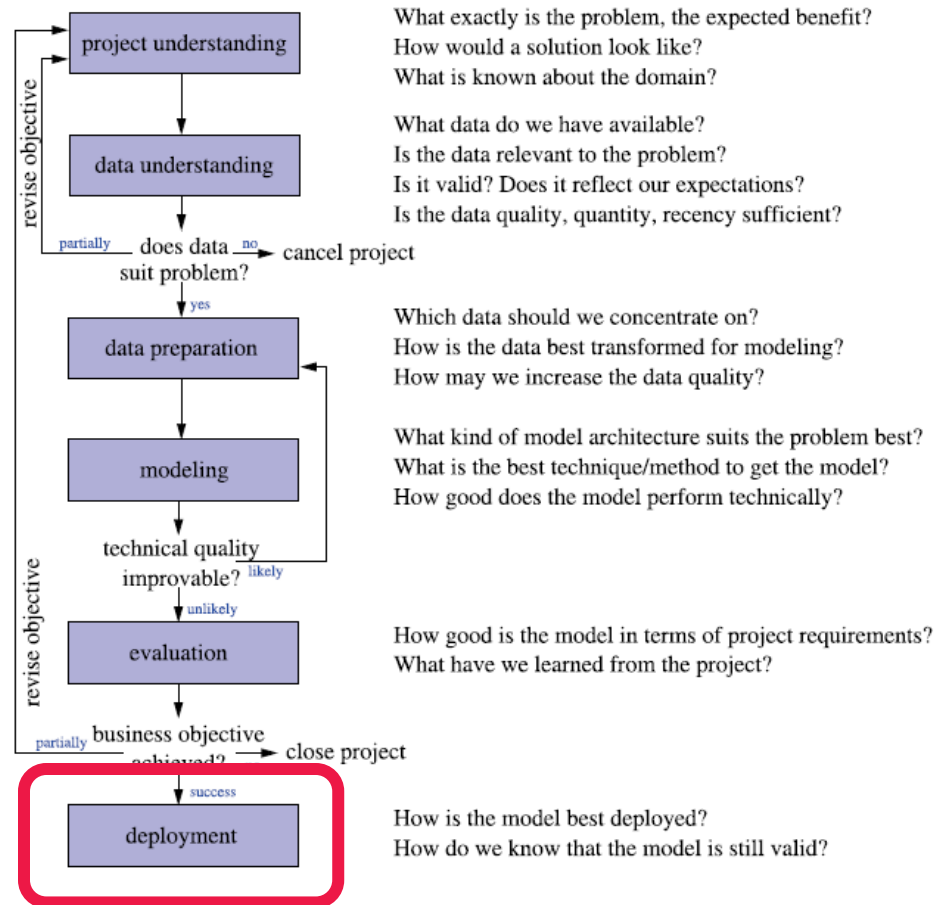
- Sample, Explore, Modify, Model, Assess

## — CRISP-DM

- Cross Industry Standard Process for Data Mining

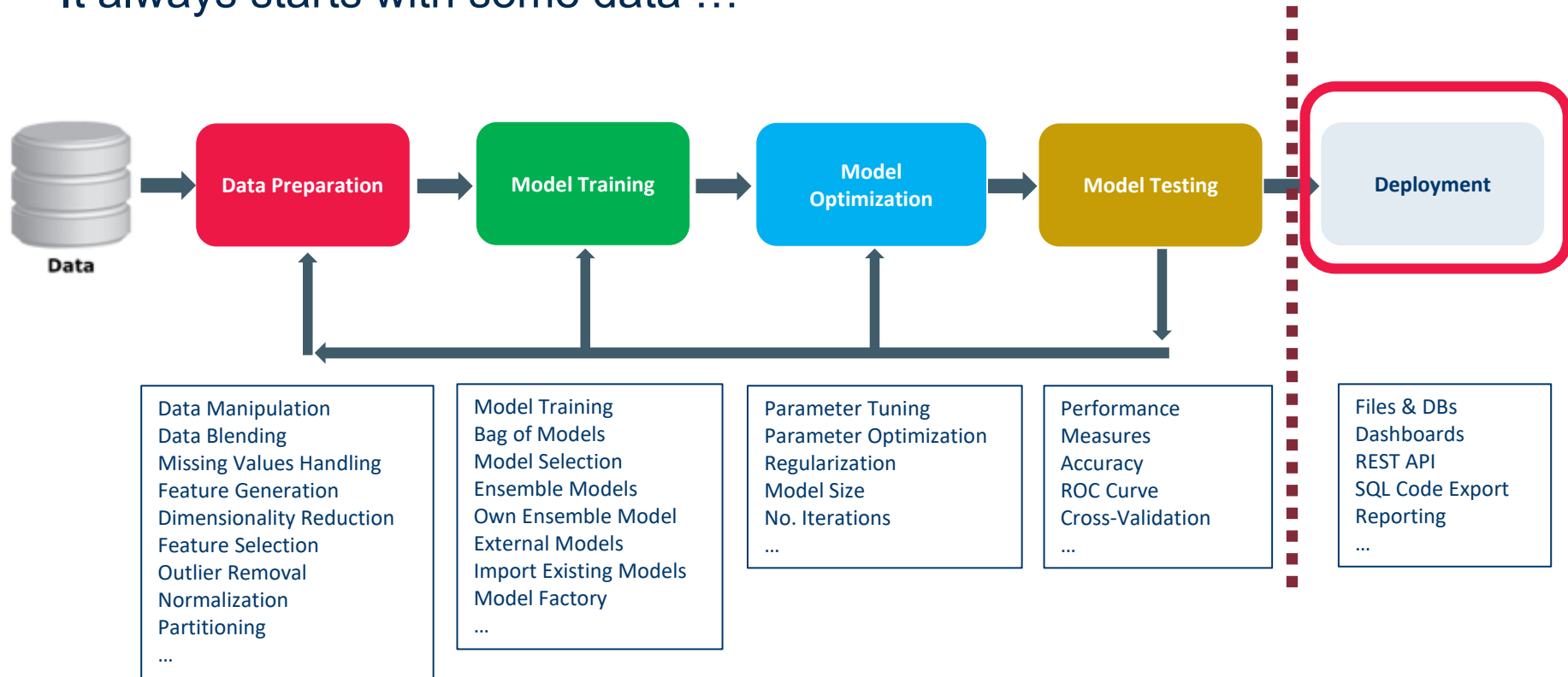
## — KDD

- Knowledge Discovery in Databases

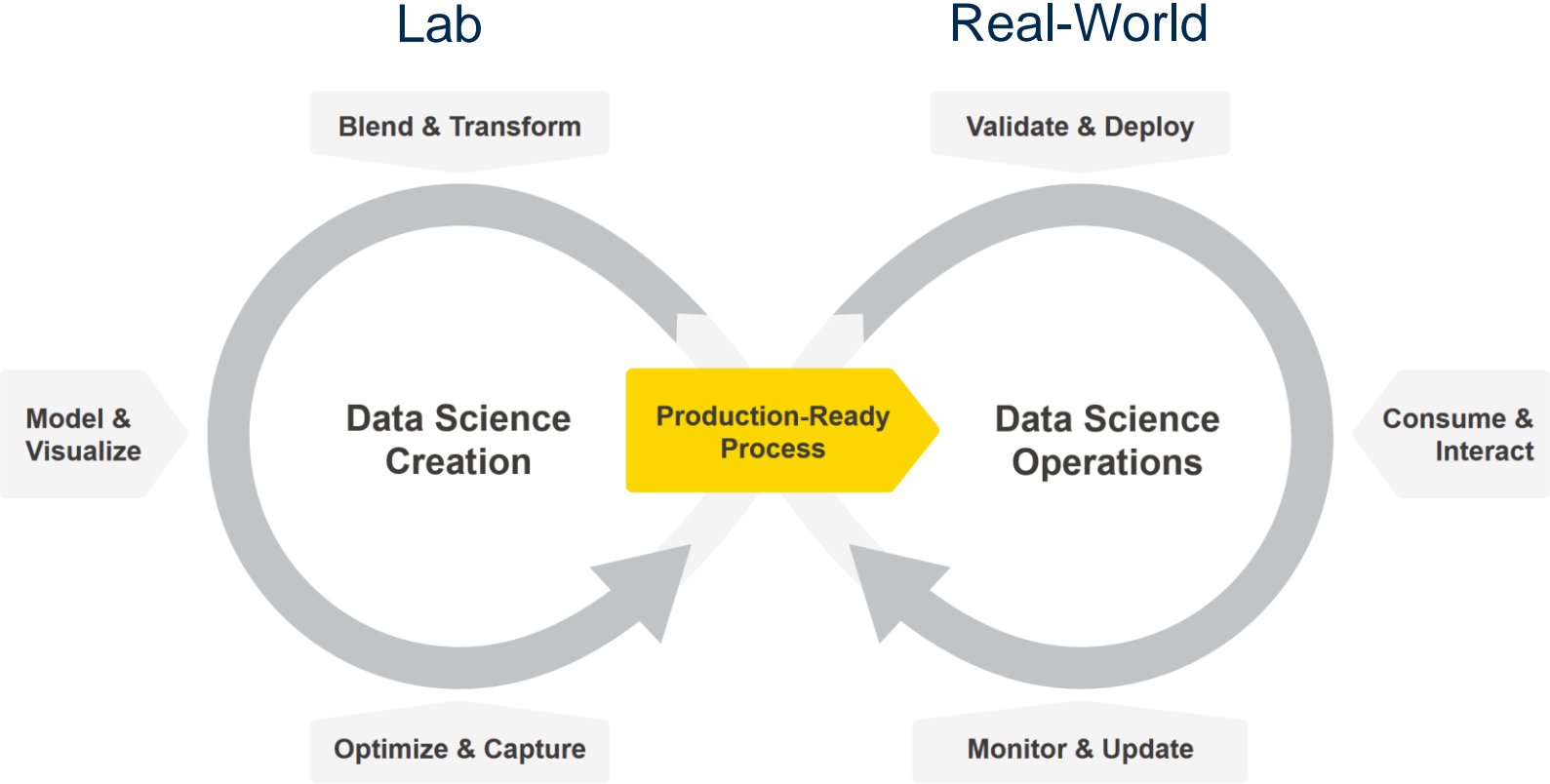


# A Classic Data Science Project

- It always starts with some data ...



# What comes after Deployment?



## What is model deployment?

- Notice the dashed line between model testing and model deployment?
- This is where the jump **from the lab to the real world** happens
- Eventually a trained model must be included in a final application to be used **by external applications and/or end users**
- The final application is the deployment application
- The step of building the application around the trained model is called **deployment**
- Notice that the deployment application must be developed and finally put into production like all pieces of software
- When the deployment application is moved into production, so is the trained model



## – Easy

- It must be easy for the application developer to include the trained model into the deployment application
- Easy to use for end users
- Easy to integrate in a Service Oriented Architecture

## – Safe

- At the same time it must be correct. For example, it must include the whole data preparation part.
- Most reasons of deployment failures are in the not faithful export of the pre-processing and post-processing steps from the training application into the deployment application.
- Think of a model trained on normalized data and of a deployment application where normalization has been forgotten.

Once in the real world, the deployment application and the trained model must oblige to the laws of IT

- **Automation**
  - On demand & scheduled execution
  - Monitoring and Updating
- **Auditing**
  - Justify decisions
  - Store previous executions
  - Reproducibility
- **Security**
  - Protection of sensitive data
  - Protection of sensitive applications
  - Versioning & Disaster Recovery

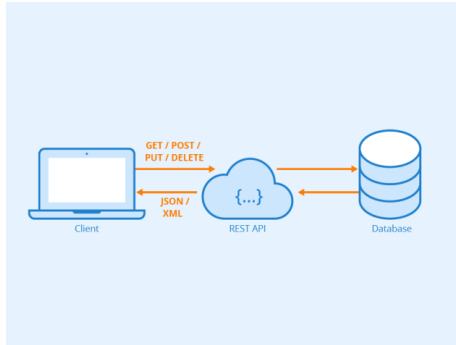
# Deployment Options

## Deployment

Usage of a trained model in an application to provide answers for a real-world use case

- In its own application
  - Easy to use for end users (as a web application)
  - Easy to integrate in a Service Oriented Architecture (as a web service)
- Consumed by external Applications
  - As a file in standard format
  - As a software library

# Deploying the ML Model



Inside its own application

In a web service

in a web application



ML model

as a file in a standard format

as a software library

```
<code>/xml version="1.0"
encoding="iso-8859-1" ?>
<language>
  <language id="fr">
    <name lang="fr">Français</name>
    <name lang="en">French</name>
    <name lang="es">Frances</name>
    <name lang="de">Französisch</name>
    <name lang="eo">Franca</name>
  </language></code>
```

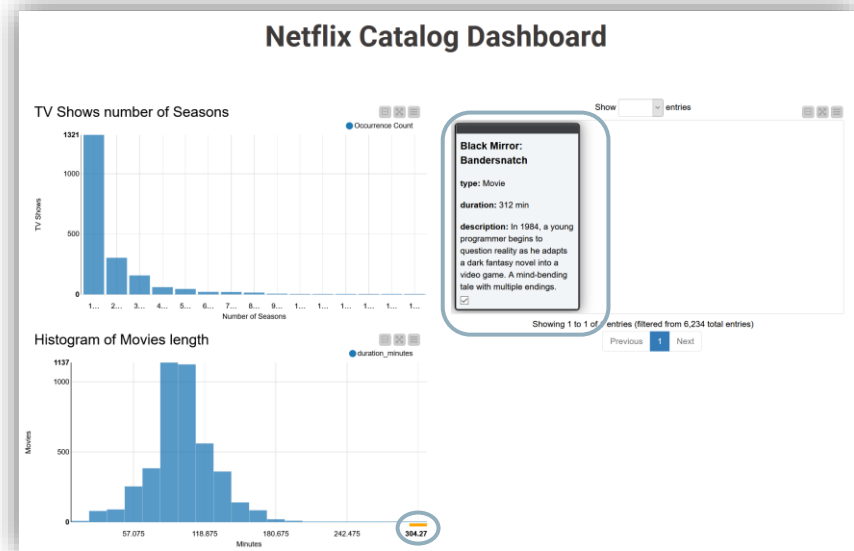
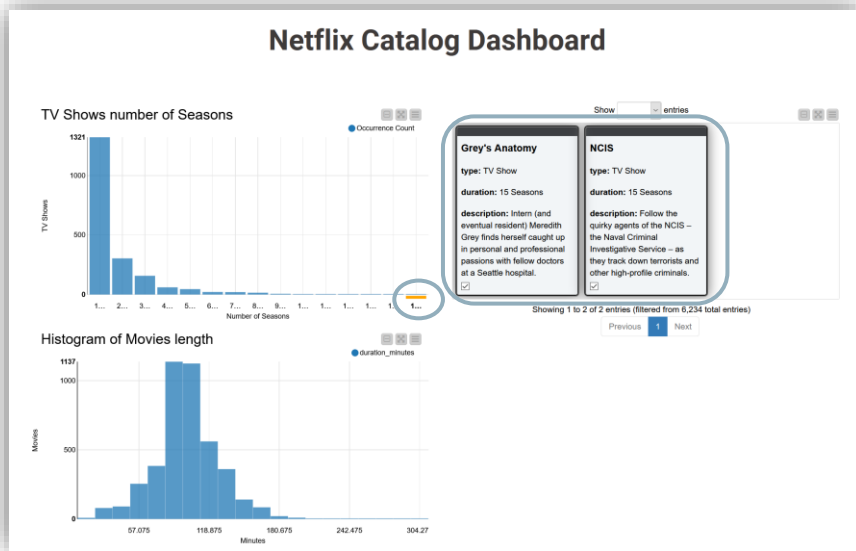
Consumed by external applications



- **Easy to use for end users**
  - If the model has been deployed into an application for end users, it must be easy to use also for non-experts and non-data-scientists kind of users
  - As a web application from a web browser
  - Hide model complexity
  - Offer touchpoints for exposed parameters
  
- **Easy to integrate in a Service Oriented Architecture**
  - As a web service
  - Via standard interfaces for web services

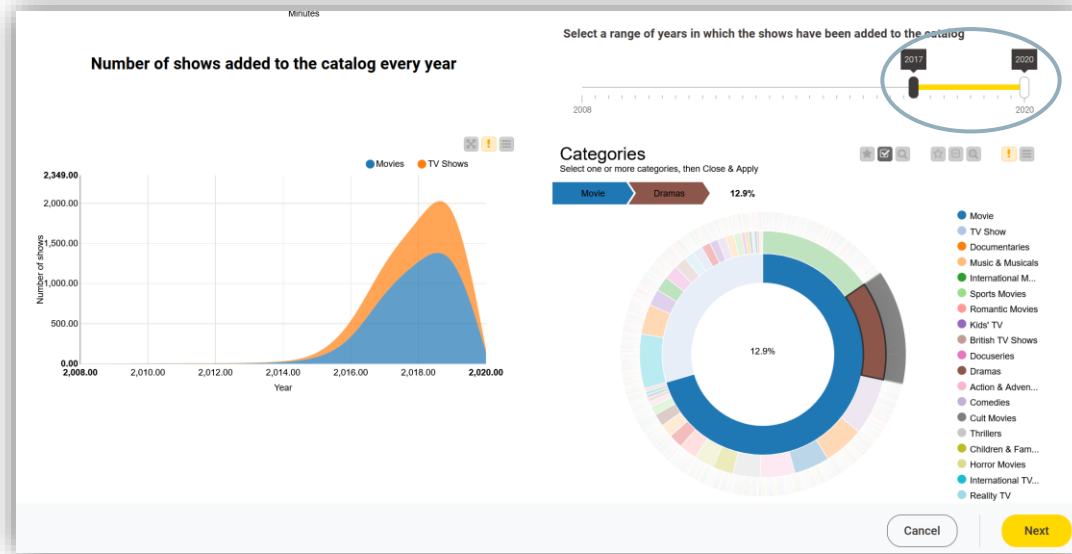
# Deployment in a web application

- One page usually includes an Interactive Dashboard to show the results
- Fast and intuitive decision support even for non expert users
- Can show model prediction and more complex interactive data visualization



# Deployment in a web application

- Interactive plots and charts
- Data selection across plots, charts, and tables
- Items such as: range slider, selection bullets, menus, ...





- One final dashboard page → to show results
  - What about having touchpoints that require end user interaction?
  - Hide complexity in automated snippets
  - Expose parameters interesting to the end users via touchpoints
- 
- **Example: Guided Automation.**
    - Train a number of models on the selected training set
    - Sequence of Touchpoints could be:



# Guided Automation: An example

## 1. Load Data

**Upload Dataset**

Upload the dataset to be used.

Upload Dataset

Upload the dataset to train the model. The dataset must be in a supported format (e.g., CSV, Excel, Hadoop) and the prediction column of each row must be specified in the 'Label' field. The data will be uploaded into the server for further processing.

## 2. Select Target

**Select Target**

Select the target column whose values should be predicted.

Select:

Label

Row ID	Workclass	Education	Num	Marital>Status	Occupation	Relationship	Race	Sex	Country	Label
Row0	State-gov	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	United-States	poor
Row1	Self-emp-not-inc	Bachelors	13	Married-cit-spouse	Exec-managerial	Husband	White	Male	United-States	poor
Row2	Private	HS-grad	9	Divorced	Handwritten-dentists	Not-in-family	White	Male	United-States	poor
Row3	Private	11th	7	Married-cit-spouse	Hardware-cleaners	Wife	Black	Male	United-States	poor
Row4	Private	Bachelors	13	Married-cit-spouse	Prof-specialty	Husband	Black	Female	Cuba	poor
Row5	Private	Masters	14	Married-cit-spouse	Exec-managerial	Wife	White	Female	United-States	poor

**Guide**

**Select Target**

To train a model, a target column must be selected. In case of classification, the target column must be a categorical variable. In case of regression, the target column must be a numerical variable. The selected target column, only valid methods for the prediction will be available.

**Details for Experts**

- If the target column is a categorical feature with 2 possible values, then it is a Binary Classification task.
- If the target column is a categorical feature with more than 2 possible values, then it is a Multiclass Classification task.
- If the target column is a numerical feature with only 2 different values, then it is a Binary Classification task.
- If the target column is a numerical feature with more than 2 and not more than 20 different values, then it is a Multiclass Regression task.

## 3. Filter Columns

**Filter Columns**

Set Column Relevance Filter

Use the slider to select a subset of columns based on their relevance. If in doubt, do not change.

418

0 100 200 300 400 500 600 700 800 900 1000

Feature	Overall Column Relevance
Age	87.13
Occupation	87.12
Education	87.04
Education-Num	87.04
Label	0
Workclass	0
Num	0
Marital>Status	0
Relationship	0
Race	0
Sex	0
Country	0
Label	0

**Guide**

**Set Column Relevance Filter**

By default, all columns will be used to train the model. However, the relevance filter can be used to select a subset of columns based on their relevance. The relevance filter is a slider that allows you to select a subset of columns based on their relevance. The relevance filter is a slider that allows you to select a subset of columns based on their relevance. The relevance filter is a slider that allows you to select a subset of columns based on their relevance.

**Column Relevance**

Column Relevance is an overall metric calculated automatically and used to determine overall Column Relevance.

The additional metrics calculated automatically and used to determine overall Column Relevance include:

- **Overall Column Relevance** is a metric used to identify and prioritize features for your model and should be increased.
- **Constant Value Test** measures how often the column contains the most common value. Columns with just a constant value also may not be informative. You should avoid using them.
- **Missing Value Test** measures the

## 4. Select Models

**Select Models**

Choose one or more machine learning models to train for your prediction task.

**Simple models**

- Naive Bayes
- Decision Tree
- Logistic Regression

**Complex models**

- Support Vector Machine
- Random Forest
- Generalized Linear Models
- Gradient Boosted Trees
- Deep Learning

**Fine-tune Model Parameters**

**Guide**

**Select Models**

Choose which models you want to train. The available models have different levels of complexity. Some complex models are designed to improve performance and generally take longer to train than more efficient models. In contrast, more complex models are capable of solving more complex problems or generally a better fit for all data points. The cost of training time and less efficient usage of resources. If you have a lot of data and you simply want to see the best performance model, use all of the complex models. However, the most with high performance. You can also choose model performance and use it to train to choose the model that best fits your data. If a complex model is available, you will see a warning for available only for the simple models and you can train them only for a subset of the available models.

If a comment indicates what you are aiming for, available only for the simple models and you can train them to choose the most efficient or easiest.

**Levels of Complexity**

- **Simple models**
  - Naive Bayes is a simple probabilistic classifier based on the Bayes' theorem.

## 5. Select Execution Engine

**Execution Settings**

Please select the desired distributed environments for the execution of the workflow.

**Available options:**

- Local execution
- Use Spark cluster if possible
- Use Apache Spark MLlib
- Use other cluster environment

**Guide**

**Execution Settings**

By default, your model process will run on your KNIME Server. There are different options to run the training of the selected models in distributed environments.

The options will depend on what you have available in your system:

- **Local execution** all parts of the workflow will run on your KNIME Server.
- **Use Spark if possible** parts of the workflow can be run parallel on your Spark cluster, e.g. the H2O Sparklyr Waterfall chart or the H2O Sparklyr Waterfall chart. The workflow will be parallelized on your KNIME Server. If available, and your cluster is available, using the right features.
- **Use Apache Spark MLlib** several model training and distribution algorithms can be executed and Spark using the Apache Spark MLlib. If relevant, any your cluster is available, training time might be reduced.
- **Use other cluster environment** workflow execution is distributed on the other cluster environment.

## 6. Show Results

**Download Models**

Here is a summary of information (performances) about the models trained based on your specifications. The first chart compares the accuracy and Area under the Curve of each model. The second chart compares the training times. The third chart compares the prediction time on a new record. The fourth chart shows the ROC (AUC). After the table to download the model parameters, a performance summary for each model is shown.

**Advanced Assessment of Models**

Each model is assessed with additional information about each model:

- **Accuracy** is a metric that measures the proportion of correct predictions.
- **Area under the Curve** is a metric that measures the performance of a model.
- **Training Time** is the time taken to train the model.
- **Prediction Time** is the time taken to predict a new record.
- **ROC (AUC)** is a metric that measures the performance of a model.

**Decision Tree**

**Gradient Boosted Trees**

**Guide**

**Download Models**

The models shown were trained based on your specifications.

Each model has its own hyperparameter selection based on either the accuracy settings or the area under the curve. The results of the model selection are shown in the table below. Compare the relevant metrics to decide which model is better.

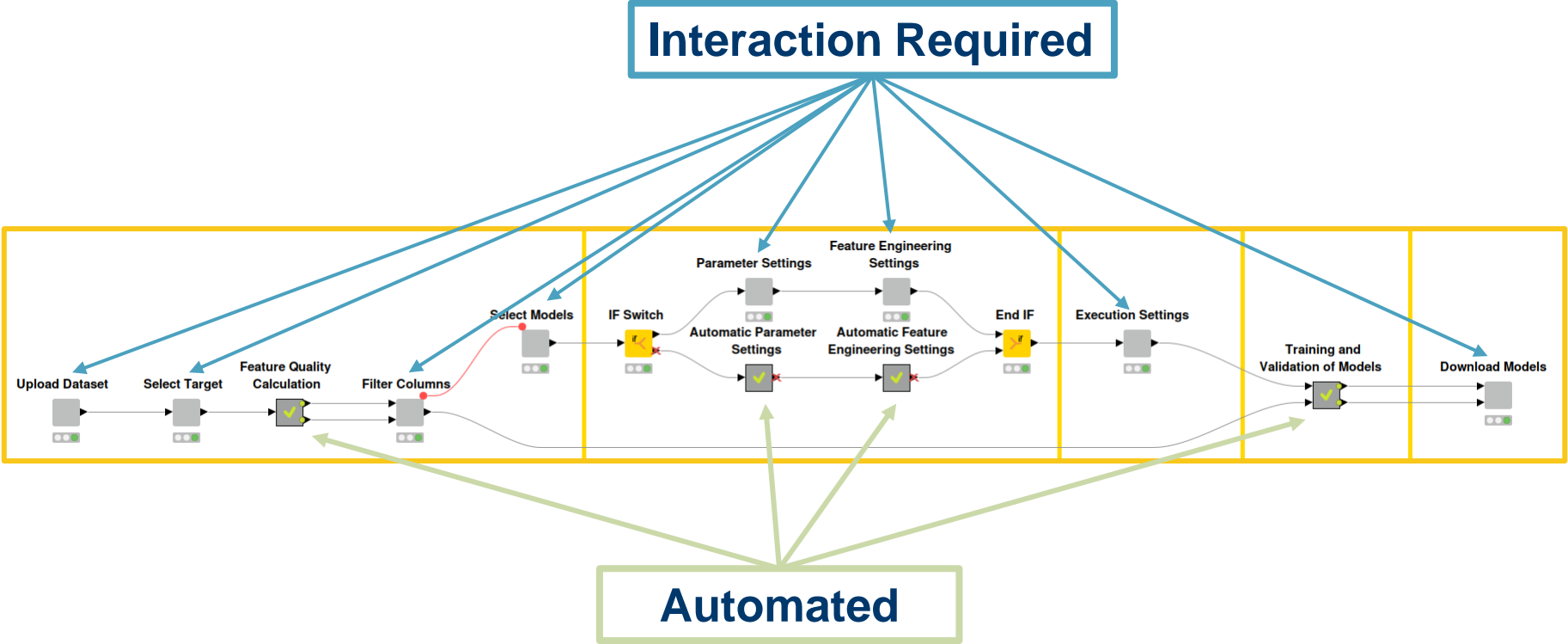
The first chart shows model accuracy and AUC. A higher accuracy is better than a lower accuracy. The amount of time needed to train a model is the result of the model's complexity. The more complex the model, the more time it will take to train. If a new model needs to be trained more often than the other models, it may be a good idea to use a simpler model. The more time it takes to train a model, the more time it will take to predict a new record. The more time it takes to train a model, the more time it will take to predict a new record. The more time it takes to train a model, the more time it will take to predict a new record.

The second chart shows the ROC curve. It provides an additional way of looking at model performance. The information provided by the ROC curve should help you decide which model is more suitable. The model with the highest accuracy is the best model. The model with the highest accuracy is the best model. The model with the highest accuracy is the best model.

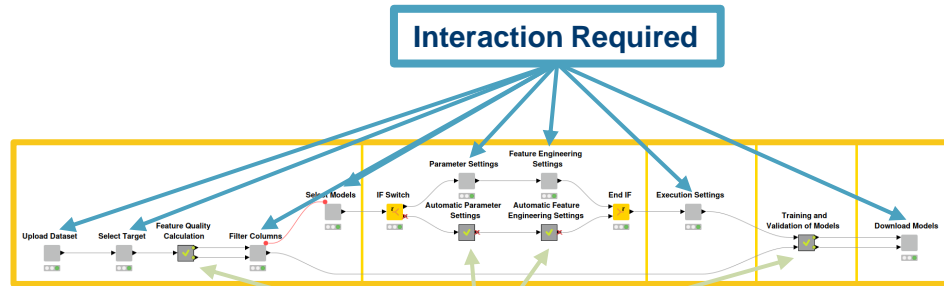
The third chart shows the prediction time. The prediction time is the time taken to predict a new record. The prediction time is the time taken to predict a new record. The prediction time is the time taken to predict a new record.

The fourth chart shows the ROC curve. It provides an additional way of looking at model performance. The information provided by the ROC curve should help you decide which model is more suitable. The model with the highest accuracy is the best model. The model with the highest accuracy is the best model. The model with the highest accuracy is the best model.

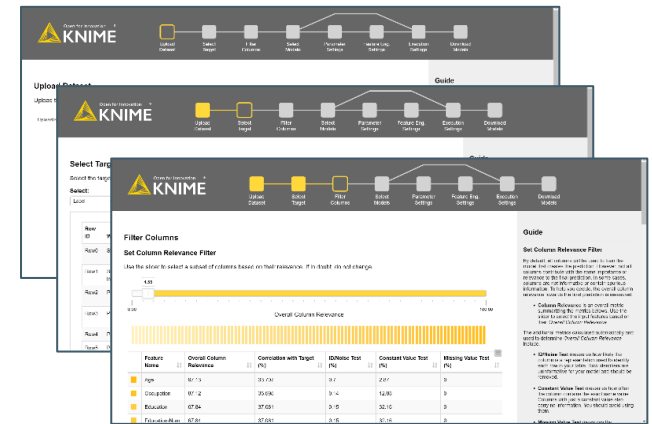
# Building a Guided Automation Workflow



# KNIME's Guided Automation: Automation + Interaction



<http://myhub.com>



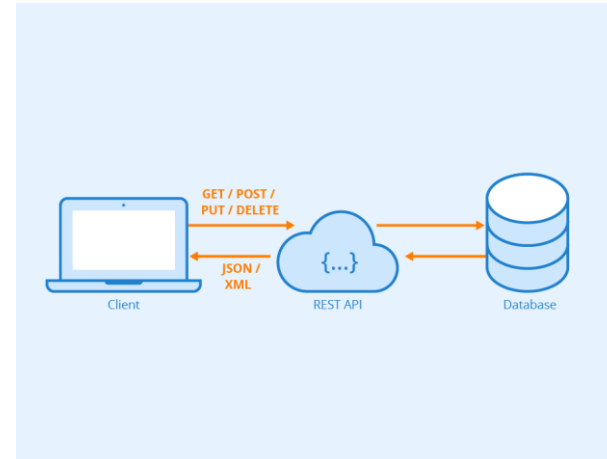
## KNIME Business Hub

analytical application  
for business users  
from web browser

- A web service provides interoperability between computer systems
  - over the internet
  - through a web technology, such as HTTP
  - to transfer machine-readable file formats such as XML and JSON.
- Web Services with REST architecture are the current state of the art
- What is a REST architecture
  - **Representational State Transfer (REST)** is a software architectural style introducing a set of constraints for web services.
  - Web services that conform to the REST architectural style, are called *RESTful* (REST) web services.
  - **REST services** allow the requesting systems to access and manipulate representations of web resources by using a **uniform** and **predefined** set of stateless operations. You cannot make up your own arbitrary set of operations, as in SOAP web services.
  - Stateless protocol and standard operations => fast execution, easy to manage

## – Operations in a REST web service (over HTTP)

- GET
- HEAD
- POST
- PUT
- PATCH
- DELETE
- CONNECT
- OPTIONS
- TRACE

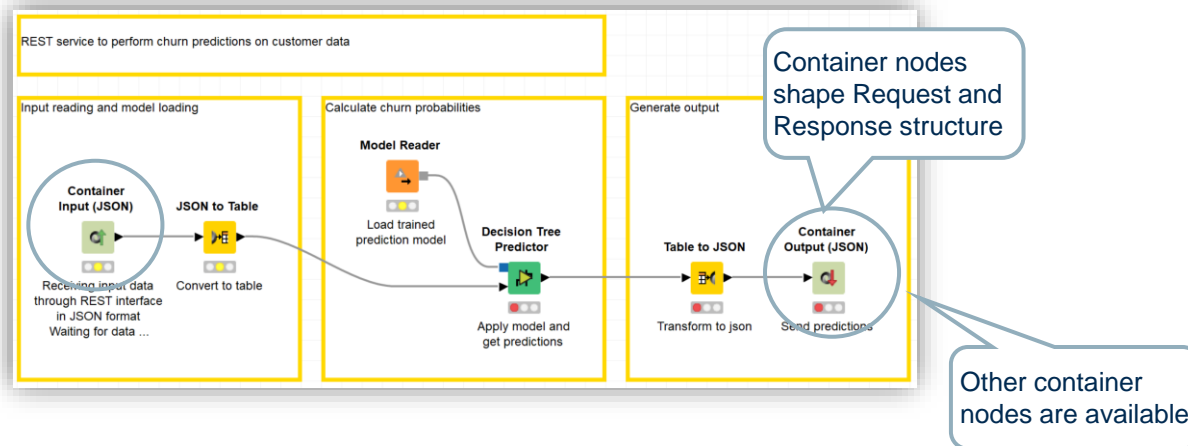


## – The Request and Response objects

- Data is exchanged via a Request object and a Response object
- The **Request object** sends data to the REST service, together with the required operation
- The **Response object** passes the result back to the calling system

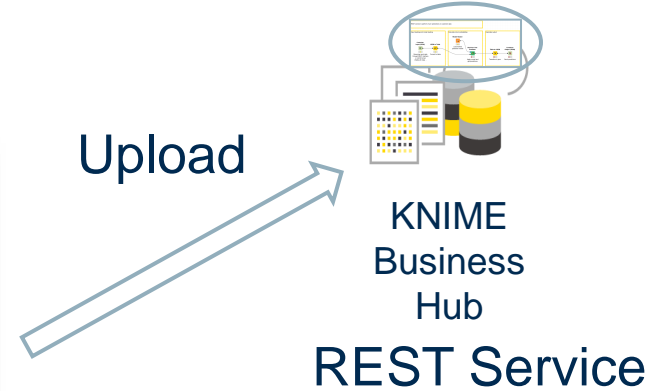
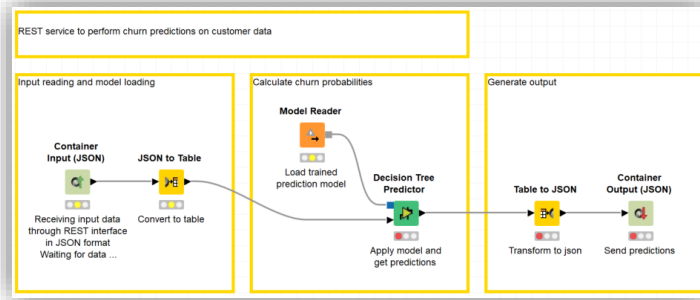
# Building a web service

- Building a REST service requires:
  - To shape the structures of the Request and Response objects
  - To enable the REST API
- Solutions:
  - Container nodes shape the Request and Response objects
  - All workflows uploaded on the KNIME Server are available as REST services



# Building a web service

1



2

## REST Service





## Deployment as a file in standard format

- Standard formats allow for external applications to consume the network/model
- **PMML**
  - **Predictive Model Markup Language (PMML)** is based on XML
  - Embeds a wide range of predictive models along with aspects of the required pre-processing
  - Can be directly loaded into database systems and applied to data tables
  - PMML works well with standard ML models (decision tree, logistic regression)
  - Representation of new complex models (ensemble, deep learning...) is problematic, either because a standard representation has not been defined or because the size of the resulting file is too large
  - Less and less used
- **ONNX**
  - ONNX = Open Neural Network for eXchange
  - Open **standard** dedicated to represent **neural networks** and **deep learning networks**
  - ONNX represented networks can then be stored into files
  - Standard ensures the portability of the represented network across systems

**Note:** Data processing (transformation/integration) must be part of the deployed model in production

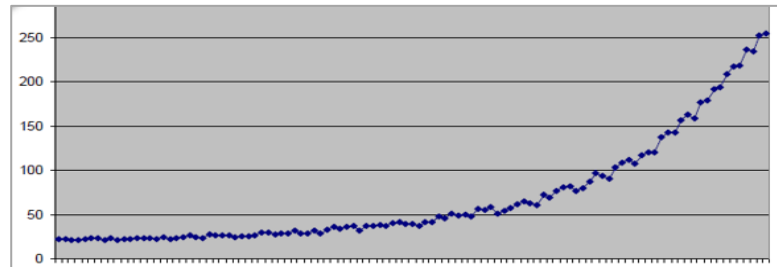
- Data Science projects often fail in deployment. Why?
- Common reasons:
  - **Bad project design:** consequences can appear only in deployment phase. For example, a feature, transformation, or a data source that is not available in production.
  - **Data leakage:** data in the test set mixes up with data in the training set. Model scores do not reflect the performances in the real-world.
  - **Dynamic domains:** Features and target variable end up having different domains in the training data vs. the real-world. New values are not handled properly.
  - **Change in Business Objectives:** During or after deployment the business objectives of the project have changed for some reason. For example, the business strategy of the company has changed.
  - **Invalidated assumptions.** What we thought it was true about the data, it is not. Maybe we did not extract a representative sample from the world data.
  - **Shift from inter- to extrapolation:** atypical data (i.e. data not used during training). What to do? Shall we stop everything?
  - **The world changes:** e.g. if new products offered or customers change habits, the data used to build and optimize the model are no longer representative of the reality

# Model Management

- The world change, the business requirement change
- **Model Management** puts in place some mechanisms to ensure that the model keeps performing as expected
- Model Management includes:
  - Model Monitoring
  - Model Update & Retraining
  - Model Factories

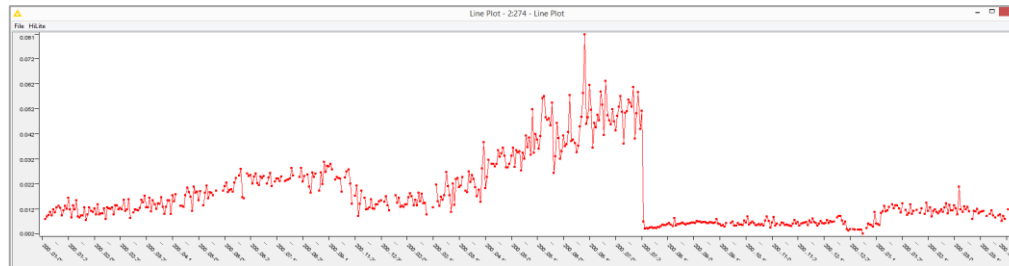
# Data Drifts and Data Jumps

- The world changes, the data change
- Data Drift (data changing slowly over time)



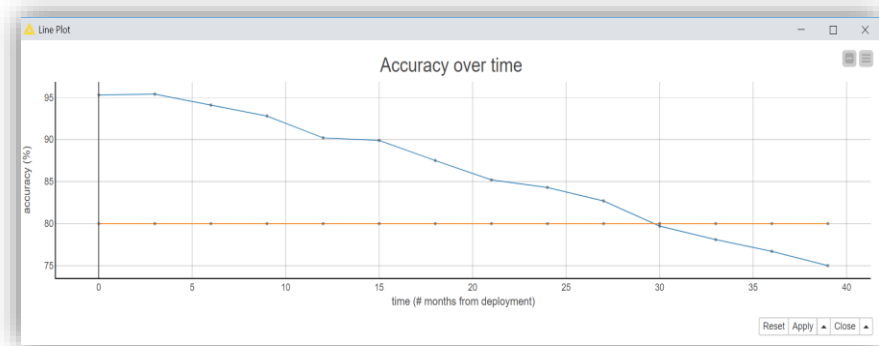
The state of a mechanical piece, the temperature going towards winter, the price of items due to inflation, etc ...

- Data Jump (Data changing suddenly at some point)



The breaking of a mechanical piece, the crash of the stock market, etc ...

- A model with an accuracy of 90% in the past can slowly (or suddenly) degrade to a much lower accuracy over time.
- This is called **Model Drift**



- Periodically check model performance
  - On which data?
  - How often is periodically?
- If model performance below threshold, retrain
  - What threshold value?

- To spot the Model Drift (due to an outdated model), you should use **recent data**
- It is of course useless to test the model on data acquired at the time when the training data were collected.
- At every run, production data are stored for monitoring purposes, till a sufficiently large dataset is collected.
- Manually annotated data are also added to test **border cases**
- The model is then tested again on this newly collected dataset.
- No action is taken if performance drops within an acceptable interval. Contrarily, actions for model retraining must be taken, if performance goes below the acceptance threshold.

- What does “periodically” mean?
- Shall I test my model performance once a week, once a month, or once a year?
- It depends on the data and on the business case:
  - Stock prices change every minute → model re-evaluation every few days
  - The taste of a customer segment will be the same for a few weeks → model re-evaluation every few months
- Same for the evaluation threshold: the value depends on the data and on the business case



## – (Automatic) Model Updating

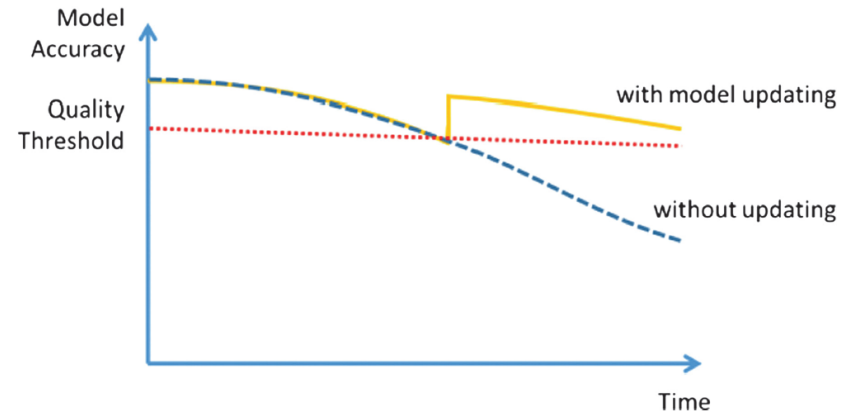
- Feed new data points to be incorporated into the model
- In this way old data are less important (are forgotten)

## – Retraining

- Use sampling to provide the right mix of past and more recent data

## – Caveats:

- Seasonality can be a problem. Specialized models or season knowledge manually injected
- Pre-existing knowledge (e.g. border case handling) better incorporated using a separate rule model instead of manual knowledge injection



## – **Model Replacement**

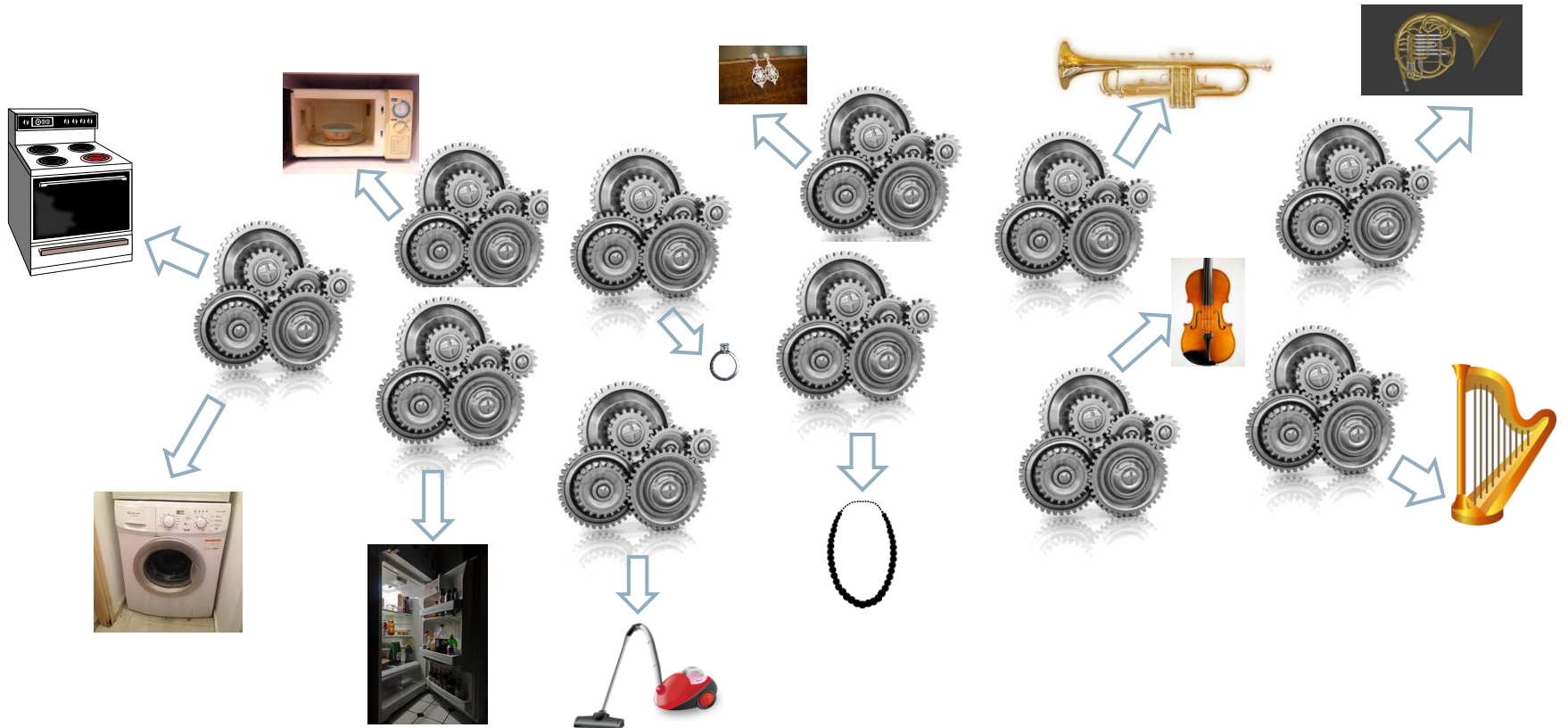
- We have retrained a new model. Are we sure it is better than the previous one?
- New model is the ***challenger***
- Former model is the ***champion***

IF challenger's performance > champion's performance THEN replace  
OTHERWISE keep champion model

## – **Caveats:**

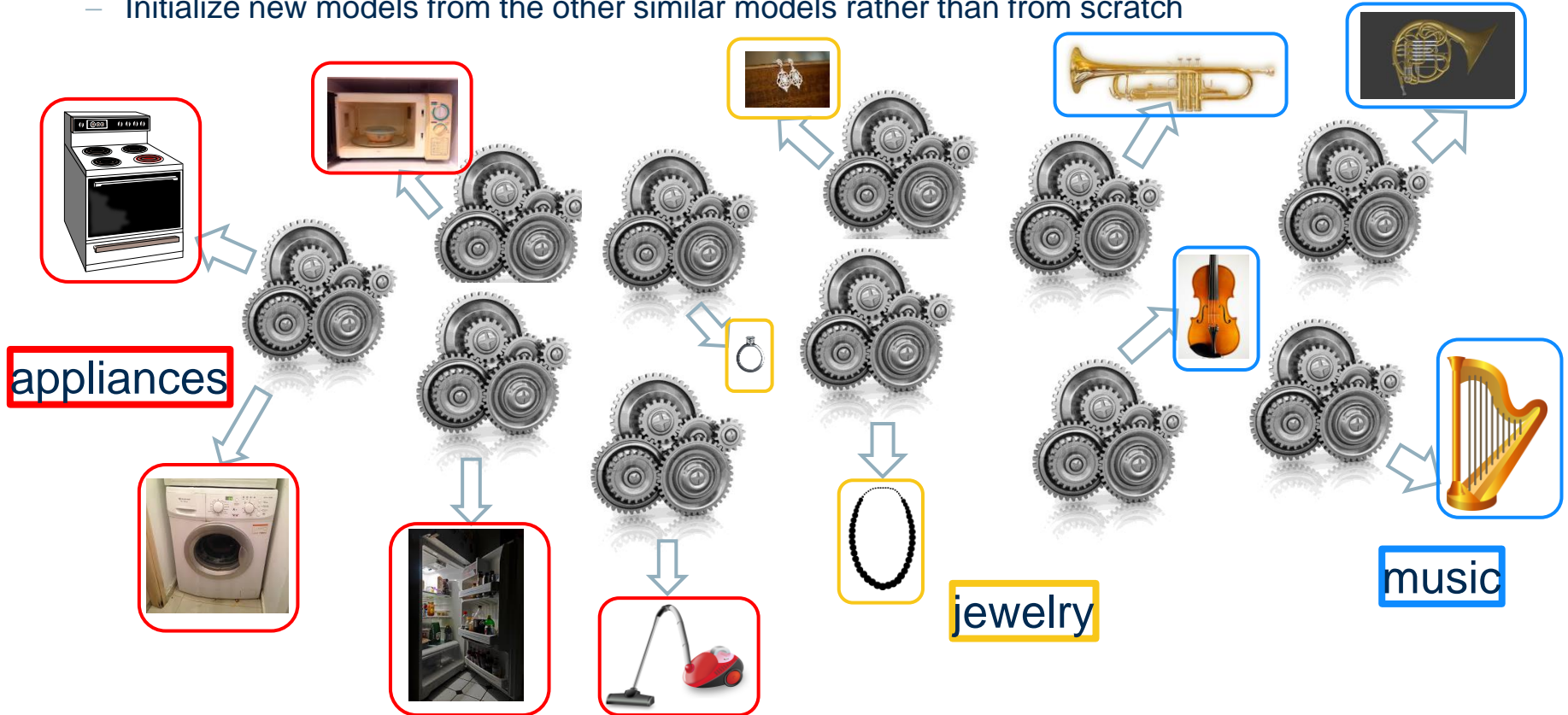
- Resources and time demanded

- Orchestration of a set of models – e.g. predicting prices



## – How to manage a set of models?

- Exploit grouping (families of similar models rather than single ones)
- Initialize new models from the other similar models rather than from scratch



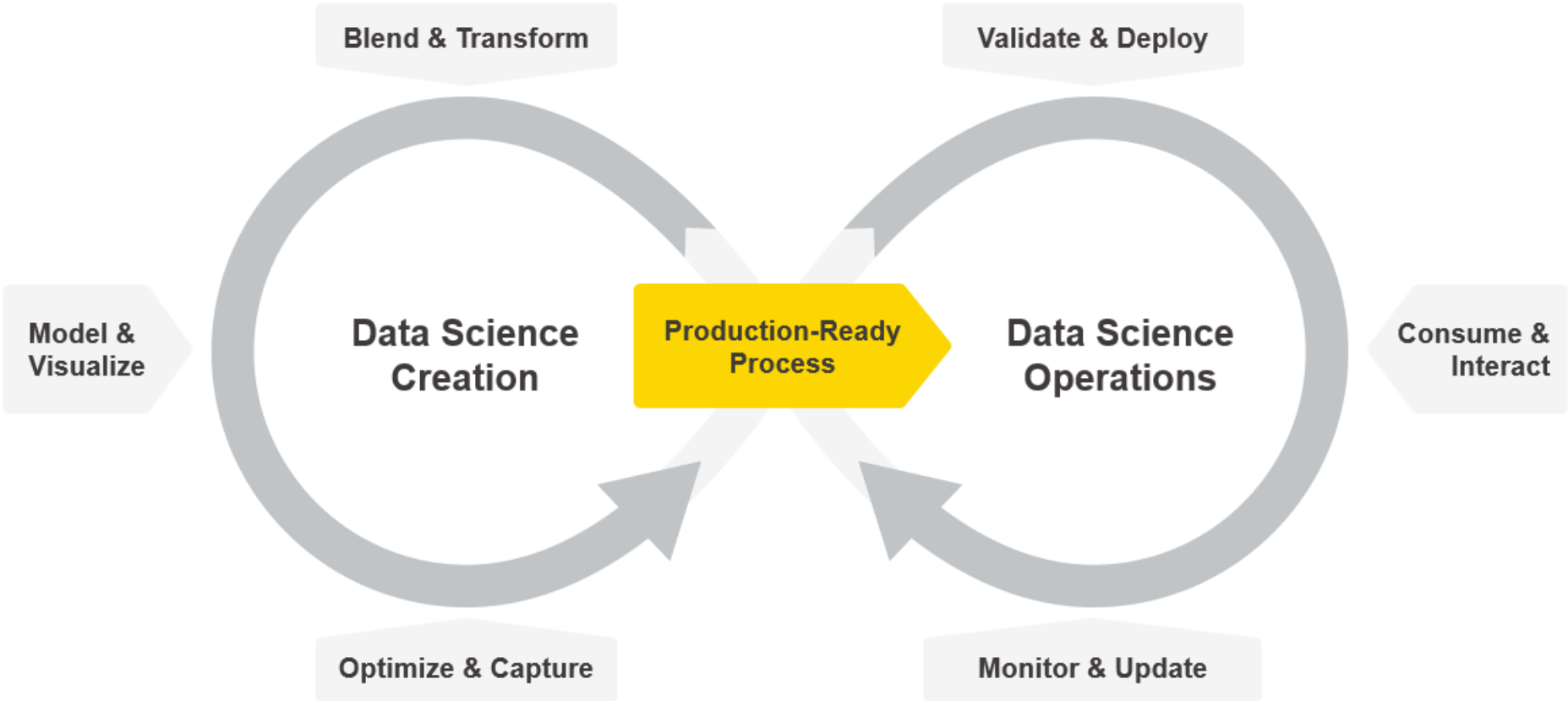
- How to communicate to the user the status of thousands of models?
  - An application for the frontend
- Who controls the process and the dependencies?
  - A separate program that handles the management process in the correct order

category	music				jewelry			appliances				
item	horn	trumpet	violin	harp	ring	Ear-rings	Neck-lace	fridge	wash. mach.	micro wave	stove	Vacuum cleaner
Threshold on accuracy	0.75	0.90	0.85	0.85	0.9	0.9	0.85	0.7	0.8	0.75	0.75	0.8
retrain	If 3 out of 4 perform below threshold				If all perform below threshold			If one performs below threshold				

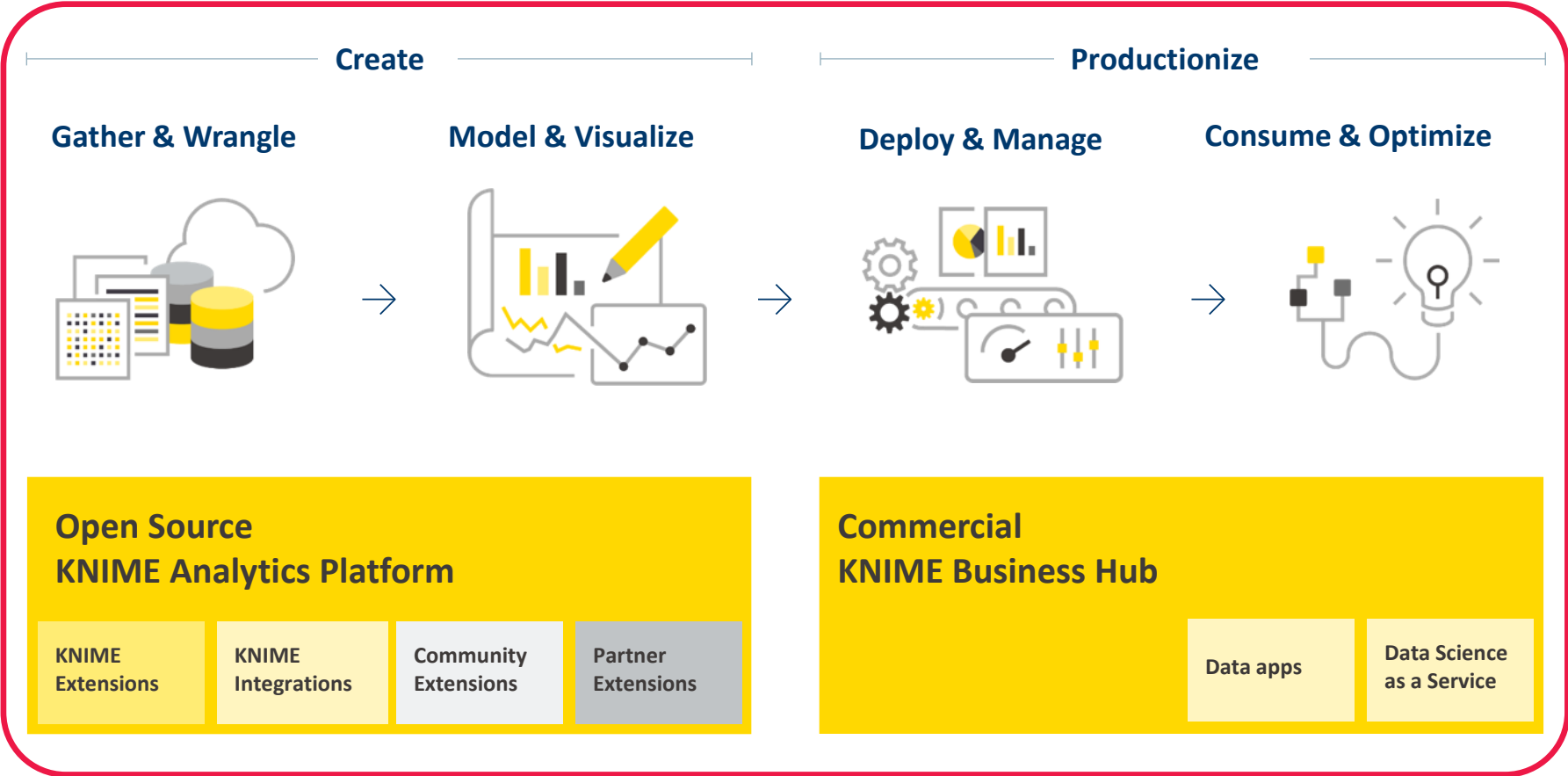
- The term **MLOps** (or **DataOps** or **DSOps**) refers to all those operations required to deploy, monitor, update/retrain a model and comply with the general company rules for auditing and data protection.
- In a sense they are similar to DevOps for software applications in a production environment, only that they deal with Machine Learning models and data science operations in general.
- MLOps Examples
  - Deployment and moving into production
  - Monitoring of Model Performance
  - Triggering of Retraining[s]
  - Storage of Information for Auditing Purposes

# Model Deployment and Management in Practice with KAP and KNIME Business Hub

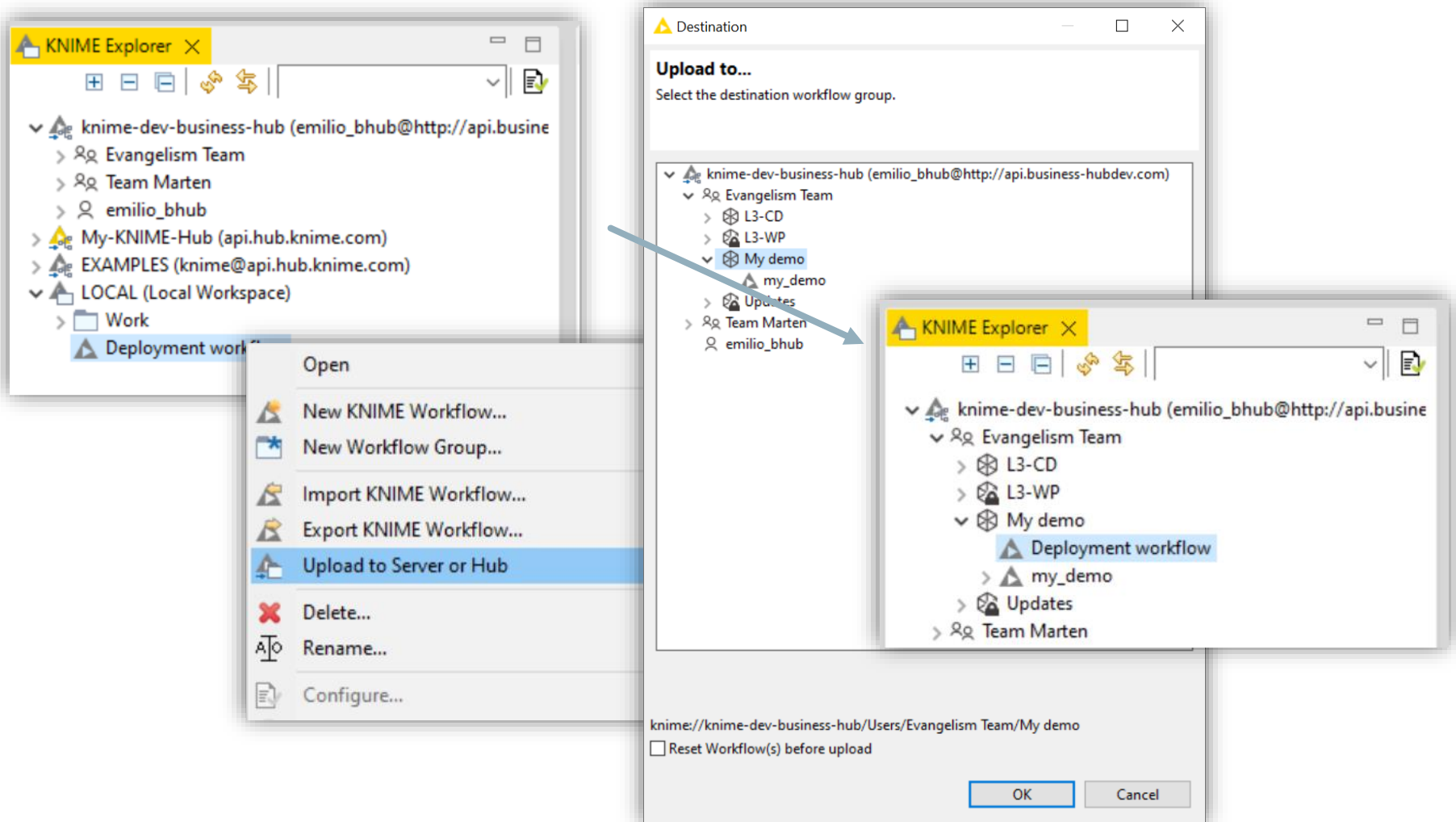
# Creating and Productionizing Data Science



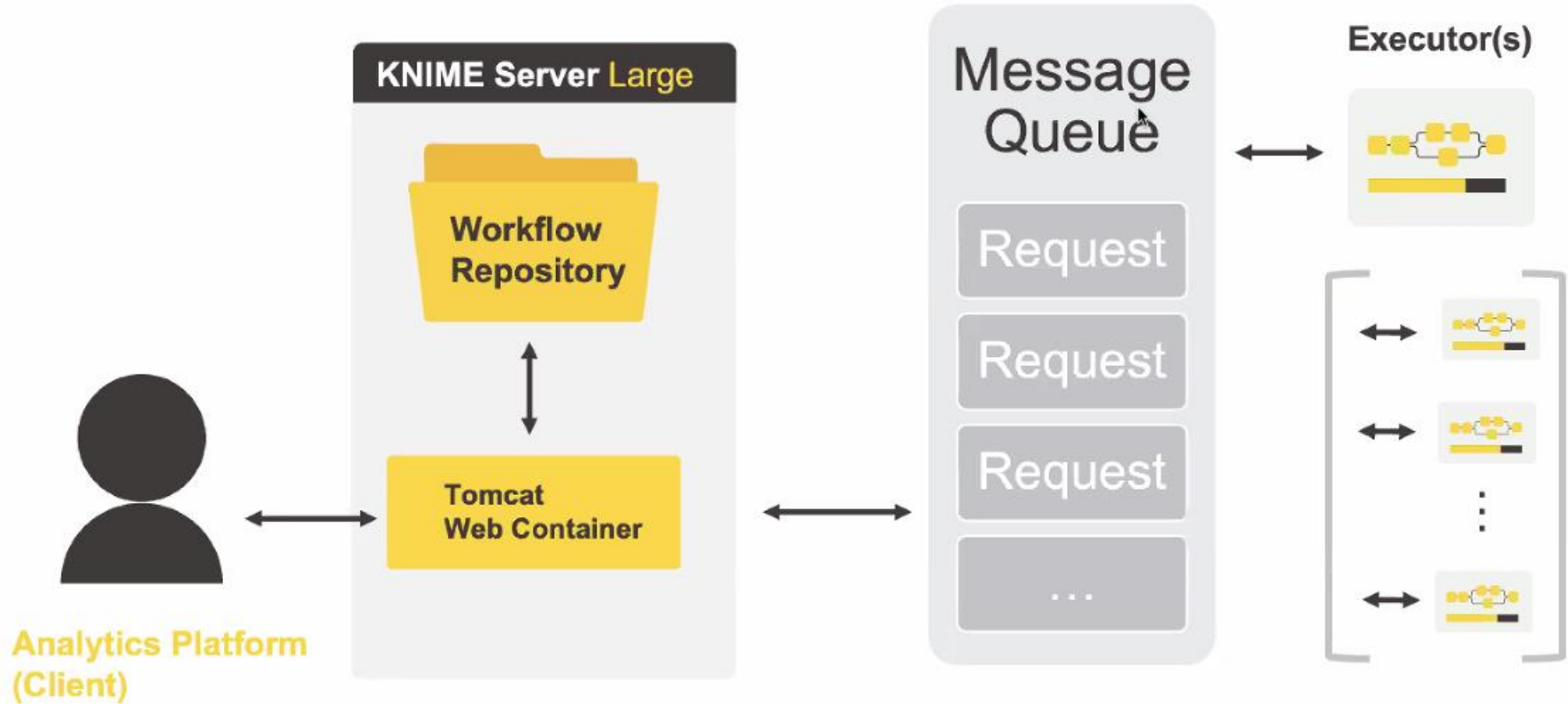





# How does deployment actually look like in KNIME?



# KNIME Server Architecture




# Usual ways to deploy a workflow on KNIME Business Hub

 **Data app** ✕

Create a data app to interact with the workflow via a user interface.

[Create data app](#)


---

 **Schedule**

Schedule your workflow to run automatically at selected times.

[Create schedule](#)

---

 **Service**

Create a service to use the workflow as an API endpoint.

[Create service](#)

# Deployment to Data App

KNIME Dev Business Hub > Evangelism Team > Spaces > Updates > 01\_Deploy\_on\_Dashboard

Workflow  
**Deployment to a Dashboard**

TheGuideBook | Deployment | Dashboard | GIS

Last edited: Jan 12, 2023 | Workflow history

Prediction dashboard. The visualization component shows prediction accuracy and data insights

**Model Reader**  
 Load trained model

**Decision Tree Predictor**

**Data upload**  
 Predict churn

**Visuali**

**Create data app**

Create a data app to interact with the workflow via a user interface.

**Deployment name**  
 Data app - Deployment to a Dashboard

**Version**  
 demo - Jan 12, 2023 10:21 AM

**Execution context**  
 Evangelism Execution Context (Default)

**Workflow actions**  
 Control basic actions of the workflow execution.  
 Enable workflow actions

**Advanced settings**  
 Change job lifecycle behaviour, timeouts, ... Set >

Cancel Create

**Predictor accuracy**

Confusion Matrix

	No (Predicted)	Yes (Predicted)	
No (Actual)	1229	59	95.42%
Yes (Actual)	63	149	70.28%
	95.12%	71.63%	

**ROC Curve**

True Positive Rate

False Positive Rate

● P(class=Yes) ● random

**Multiple features relations**

Parallel Coordinates Plot

**Two features visualization**

Scatter Plot

account\_length

total\_day\_charge

Show 10 entries Search:

RowID	state	account_length	area_code	phone_number	international_plan	voice_mail_plan	number_vmai
Row2	39	74	415	916	0	0	0
Row3	12	168	408	1854	0	0	0
Row5	45	76	510	1530	0	1	33

# Deployment by Scheduling Automation

The screenshot displays the KNIME Hub interface for a workflow named 'my\_demo'. The workflow diagram shows three nodes: 'CSV Reader', 'Component', and 'Automated Visualization'. The 'Create schedule' dialog box is open, allowing the user to configure automatic execution. The dialog includes fields for 'Deployment name' (Schedule - my\_demo), 'Version' (version1 - Jan 6, 2023 5:33 PM), 'Execution context' (Evangelism Execution Context), and 'Schedule options' (Initial execution date and time: 2023-01-12 10:18:00). There are also checkboxes for 'Repeat every' and 'Enable workflow actions'.

**KNIME** Open for Innovation  
Hub

KNIME Dev Business Hub > Evangelism Team > Spaces > My demo > my\_demo

Workflow  
**my\_demo**

Last edited: Jan 6, 2023 Workflow history

**CSV Reader** → **Component** → **Automated Visualization**

**Used extensions & nodes**

Extensions Nodes

Created with KNIME Analytics Platform version 4.7.0

**KNIME Base nodes** KNIME AG, Zurich, Switzerland  
Version 4.7.0

**Create schedule**

Schedule your workflow to run automatically at selected times.

**Deployment name**  
Schedule - my\_demo

**Version**  
version1 - Jan 6, 2023 5:33 PM

**Execution context**  
Evangelism Execution Context

**Schedule options**  
Define when the workflow should be executed.

**Initial execution**  
2023-01-12 10 : 18 : 00

Repeat every

**Workflow actions**  
Control basic actions of the workflow execution.

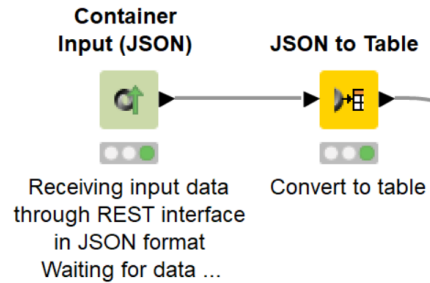
Enable workflow actions

Cancel Create

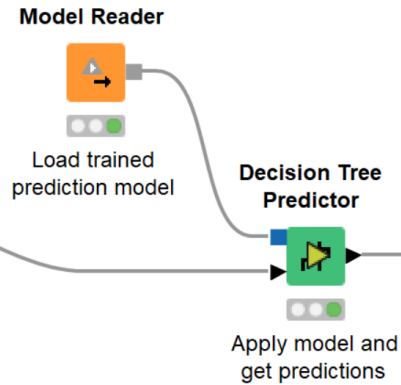
# Deployment as a REST Service

REST service to perform churn predictions on customer data

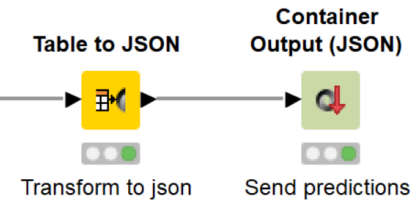
Input reading and model loading



Calculate churn probabilities



Generate output



manually

- Deployment to Data App
  - Deployment by Scheduling Automation
  - Deployment as a REST Service
- 
- Integrated Deployment

Deployment can be a repetitive task:

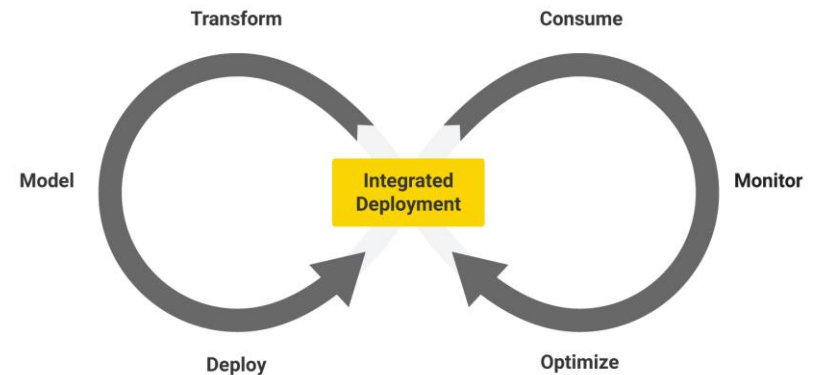
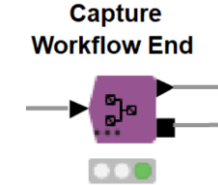
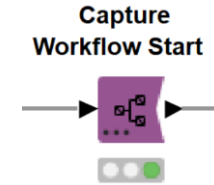
- *Monitor & Update* (from cycle)

Automating deployment of any of the above, especially **REST Service**



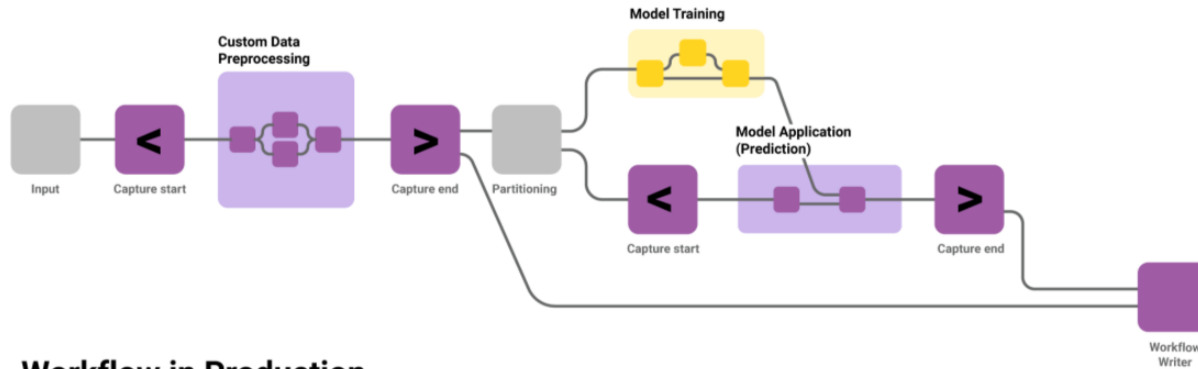
# Integrated Deployment

- Build an optimal model
- Isolate core parts of the workflow (preprocessing, model building...) with the special nodes *Capture Workflow Start* and *Capture Workflow End* from the training workflow
- Export the extracted pieces to build the deployment workflow

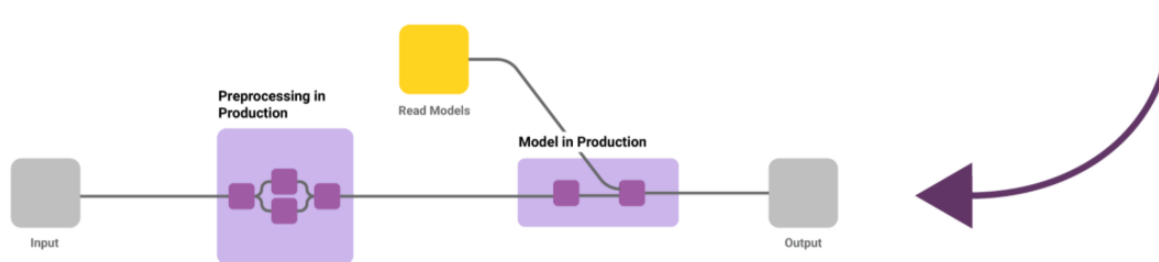


# Integrated Deployment

## Creating Prediction Model



## Workflow in Production



- Automatically build and deploy deployment workflows
- Mostly used to automatically capture and deploy a model as REST API from the workflow which trains and validates the model

# Thank you

For any questions please contact: [education@knime.com](mailto:education@knime.com)