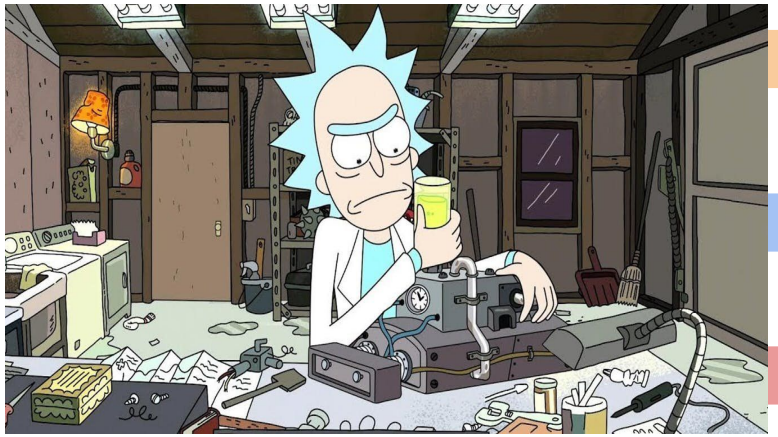
perimeterx

Boosting Trees in Production

DataTalks #9 *Meetup*, January 2019

Motivation

RESEARCH

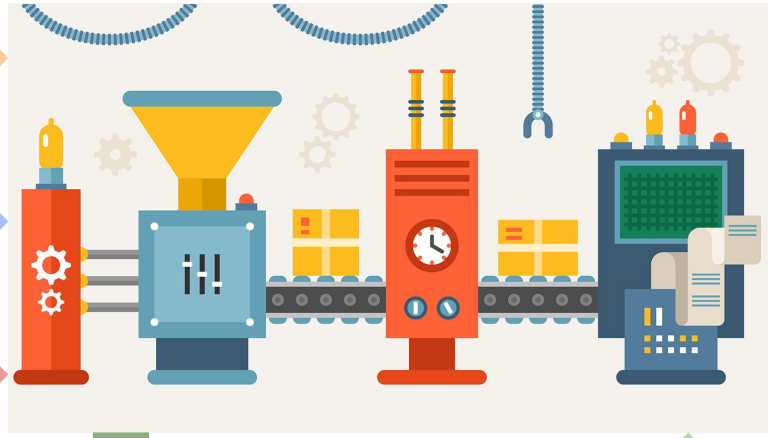


Changing model features

Changing model meta-parameters

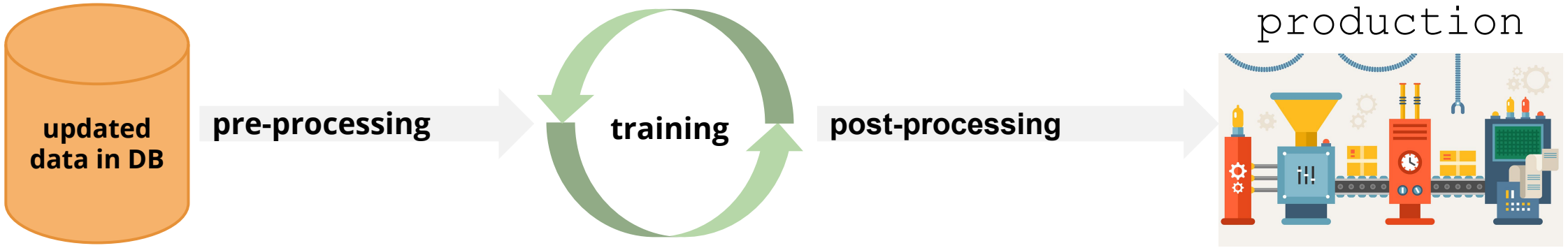
Changing model type

production



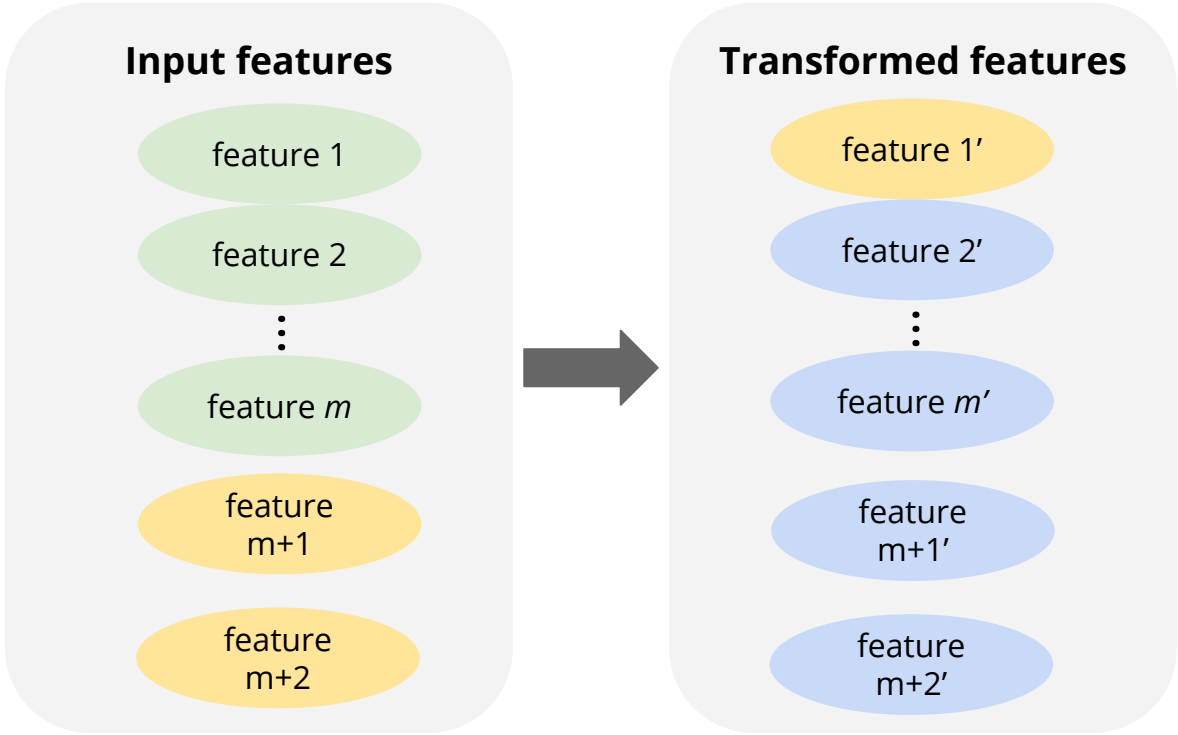
periodic training

Periodic Training

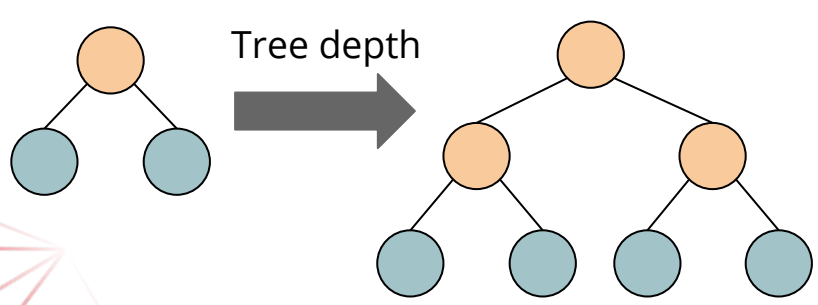


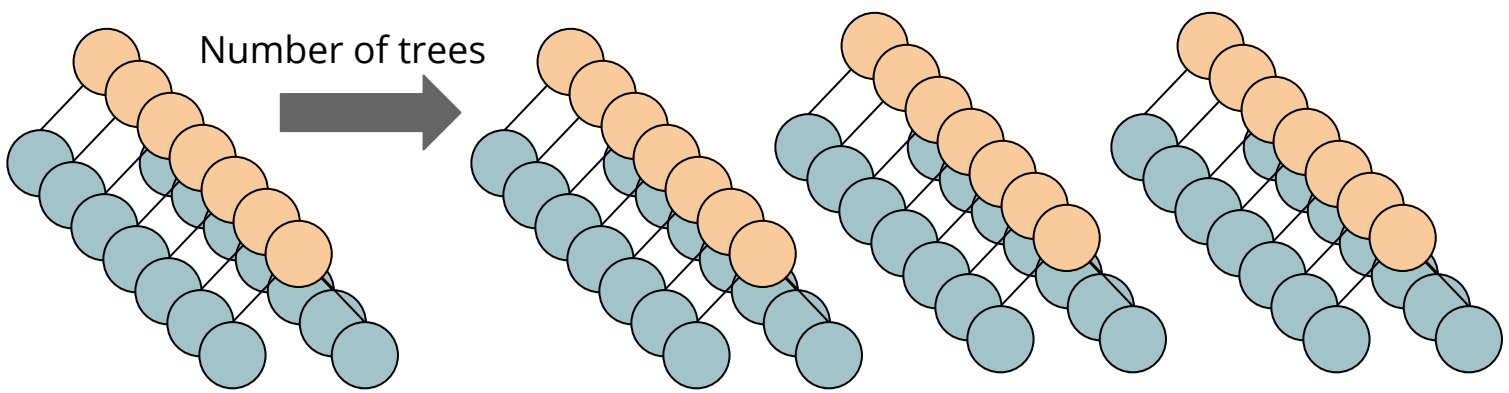
Changing Model Features

pre-processing



Changing Model Meta-Parameters





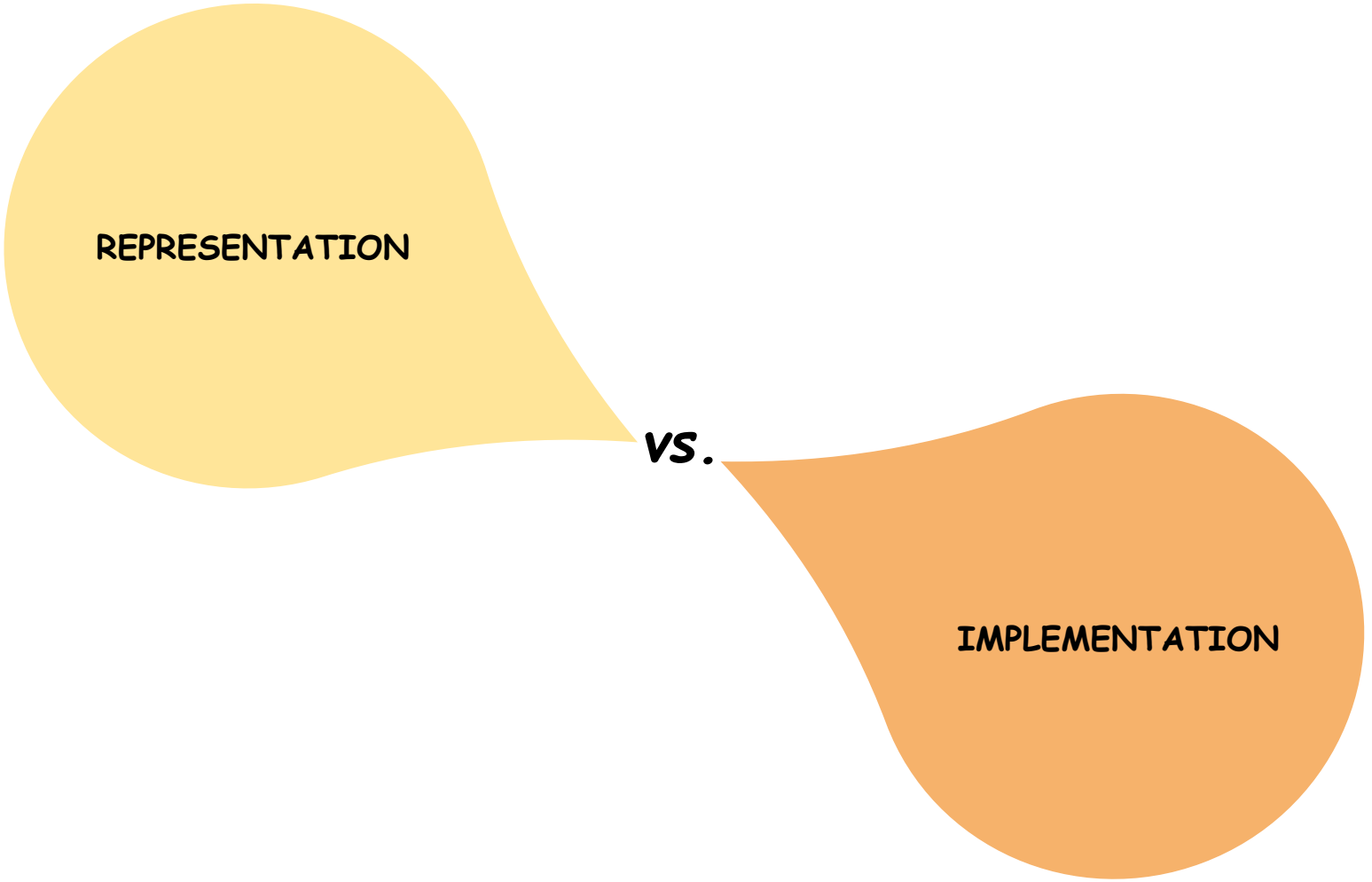
Changing Model Type

dmlc XGBoost

LightGBM

Microsoft

Approaches



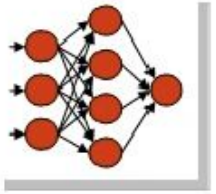
Representation

PMML
Predictive Model Markup Language

DATA MINING GROUP

PFA
Portable Format for Analytics

```
<NeuralNetwork modelName="Iris_NN" functionName="classification" activationFunction="tanh">
    <MiningSchema>
        <MiningField name="sepal_length" />
        <MiningField name="sepal_width" />
        <MiningField name="petal_length" />
        <MiningField name="petal_width" />
        <MiningField name="class" usageType="predicted" />
    </MiningSchema>
    <NeuralInputs>
        <NeuralInput id="0">
            <DerivedField dataType="double" optype="continuous">
                <FieldRef field="derived_sepal_length" />
            </DerivedField>
        </NeuralInput>
        ...
    </NeuralInputs>
    <NeuralLayer numberOfNeurons="?">
        <Neuron id="4" bias="-3.1808306946637">
            <Con from="0" weight="0.119477686963504" />
            <Con from="1" weight="-1.97301278112877" />
            <Con from="2" weight="3.04381251760906" />
            <Con from="3" weight="3.15301106009219" />
        </Neuron>
        ...
    </NeuralLayer>
    ...
</NeuralNetwork>
```



PFA document (YAML)

```yaml
input: {type: array, items: double}
output: string
cells:
  clusters:
    type:
      type: array
      items:
        type: record
        name: Cluster
        fields:
          - {name: center, type: {type: array, items: double}}
          - {name: id, type: string}
    init:
      - {id: one, center: [1, 1, 1, 1, 1]}
      - {id: two, center: [2, 2, 2, 2, 2]}
      - {id: three, center: [3, 3, 3, 3, 3]}
      - {id: four, center: [4, 4, 4, 4, 4]}
      - {id: five, center: [5, 5, 5, 5, 5]}
action:
  attr:
    model.cluster.closest:
      - input
      - cell: clusters
      - params:
          - x: {type: array, items: double}
          - y: {type: array, items: double}
        ret: double
        do:
          metric.euclidean:
            - fcn: metric.absDiff
            - x
            - y
  path: [[id]]
```

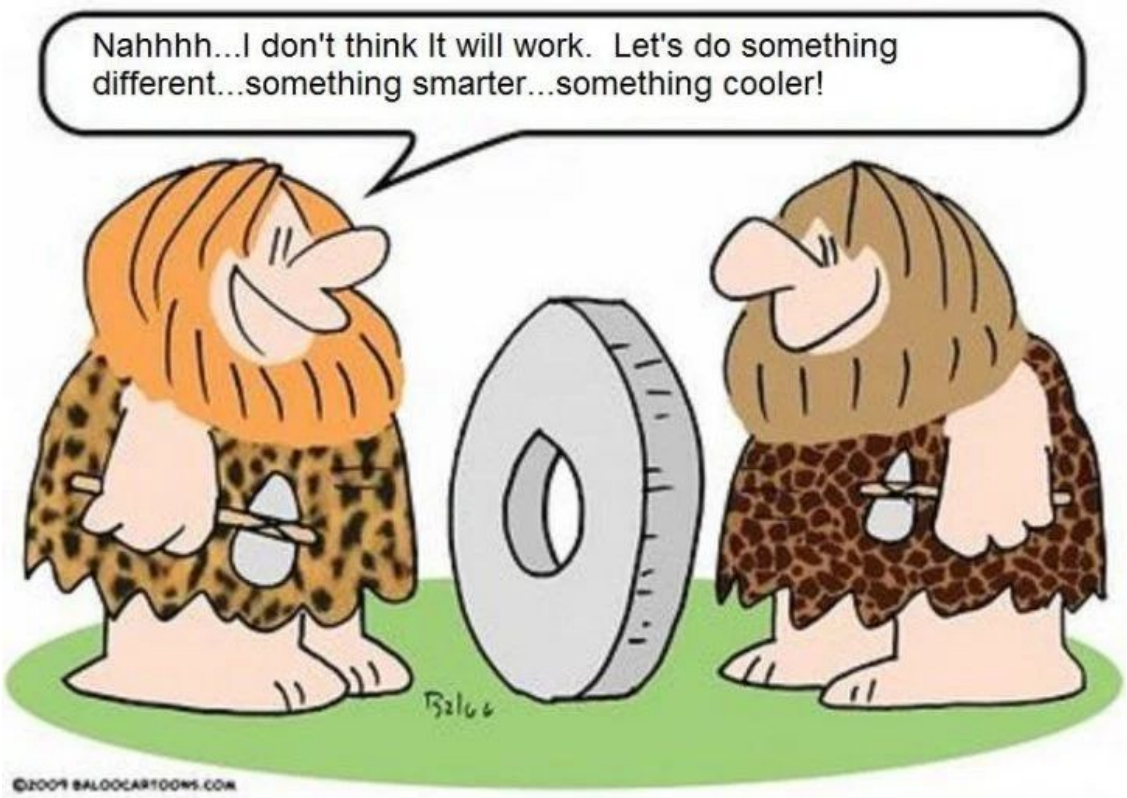
Implementation

PredictionIO

TensorFlow Serving

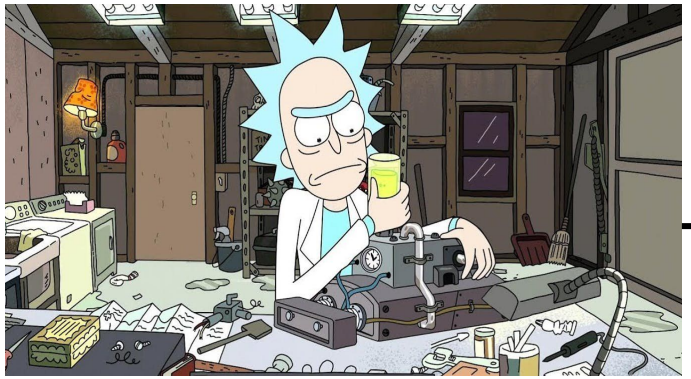
MML (Microsoft)
Spark Serving

Our In-House Solution



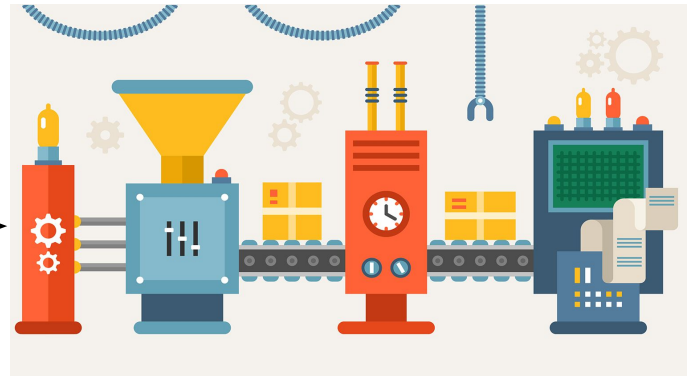
Our In-House Solution - Chronicle

RESEARCH



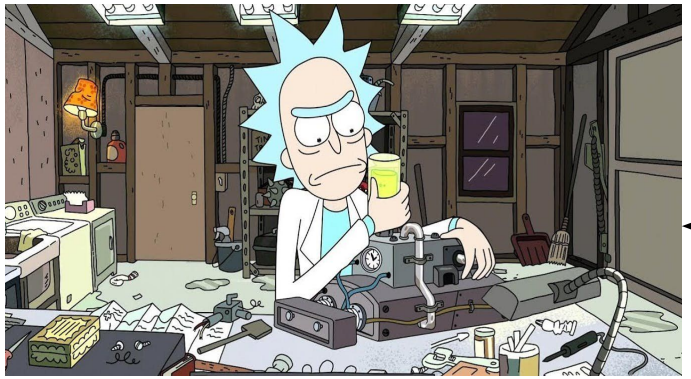
We have a nice model based on *XGBoost*
We just need you to run it in production...

production



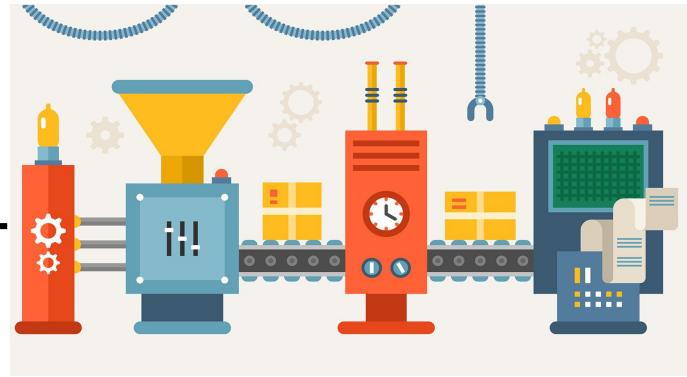
Our In-House Solution - Chronicle

RESEARCH



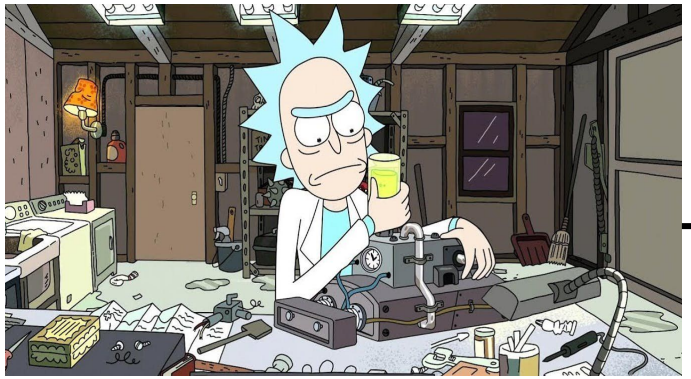
ok, but we implement things in *Go*...

production



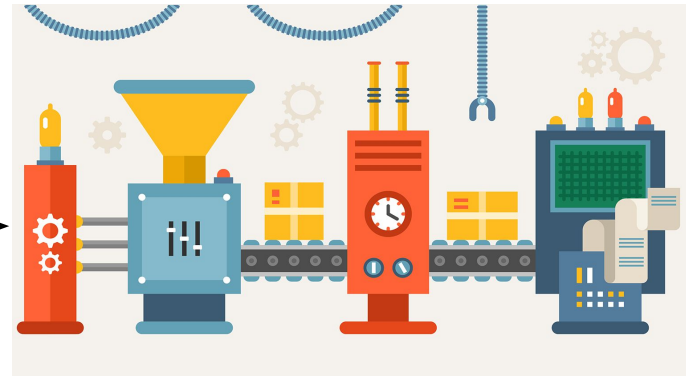
Our In-House Solution - Chronicle

RESEARCH



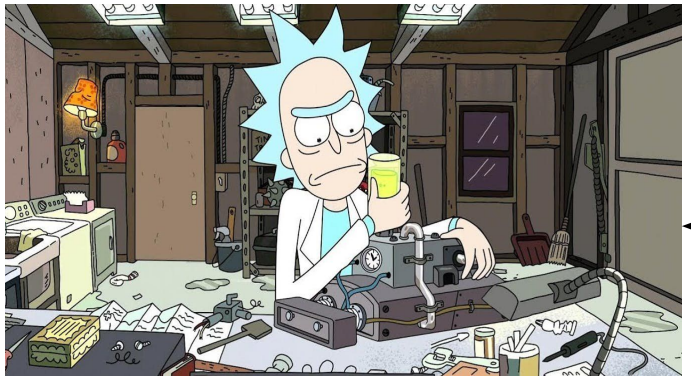
No problem, here - take this PMML and use it in you *Go*

production



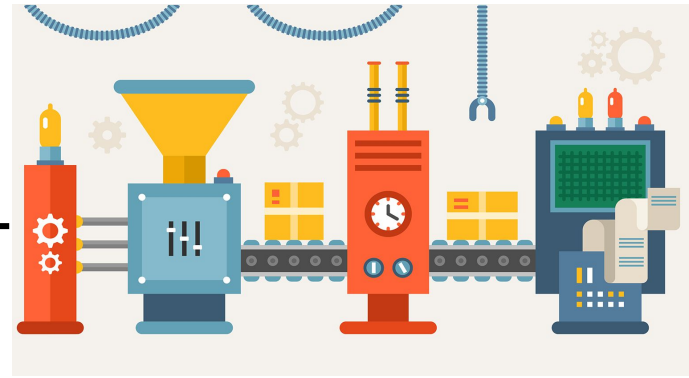
Our In-House Solution - Chronicle

RESEARCH



ok, we found some basic third-party code to do it

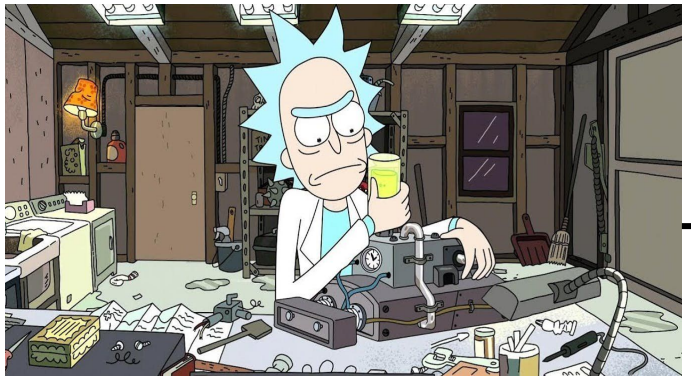
production



perimeterx

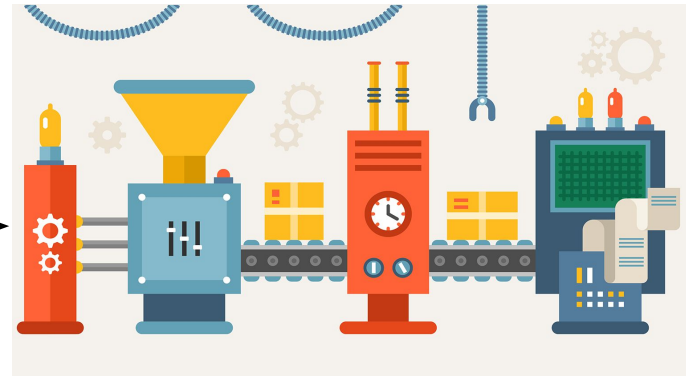
Our In-House Solution - Chronicle

RESEARCH



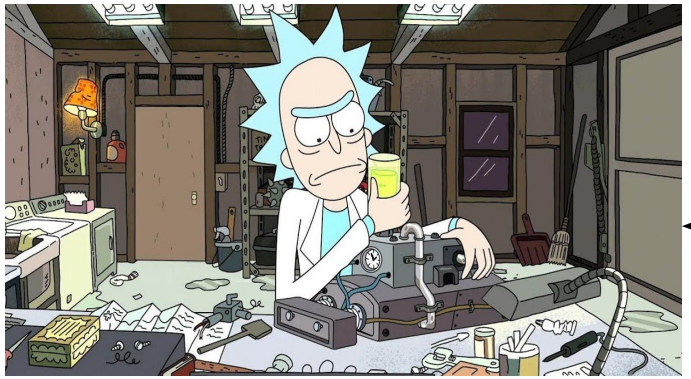
WAIT! PMML doesn't support all we need for preprocessing

production



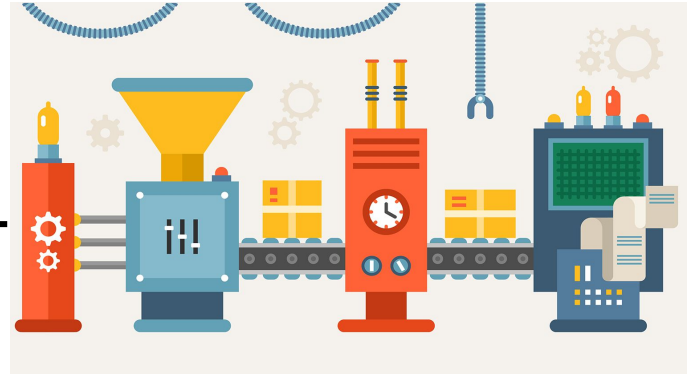
Our In-House Solution - Chronicle

RESEARCH



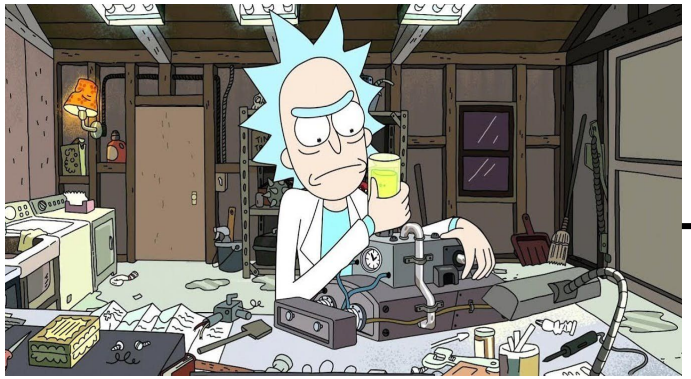
okey... how can you export the model

production



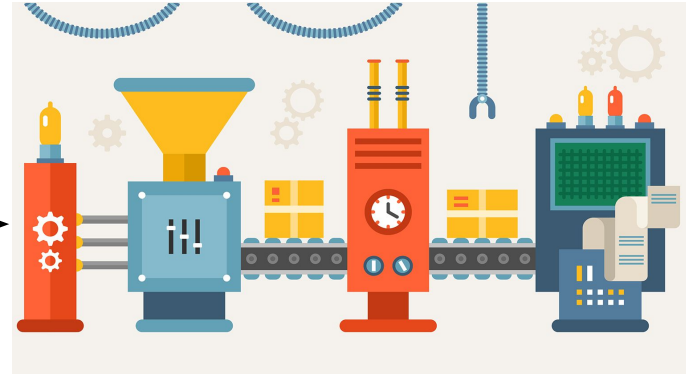
Our In-House Solution - Chronicle

RESEARCH



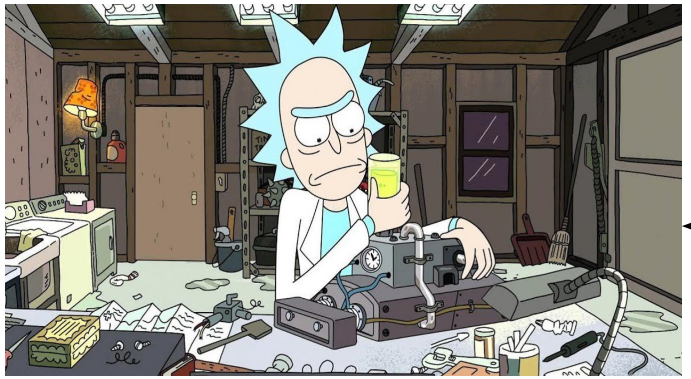
Easy, take this JSON.
Also, we need these feature transformations

production

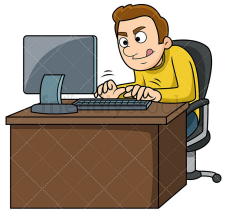


Our In-House Solution - Chronicle

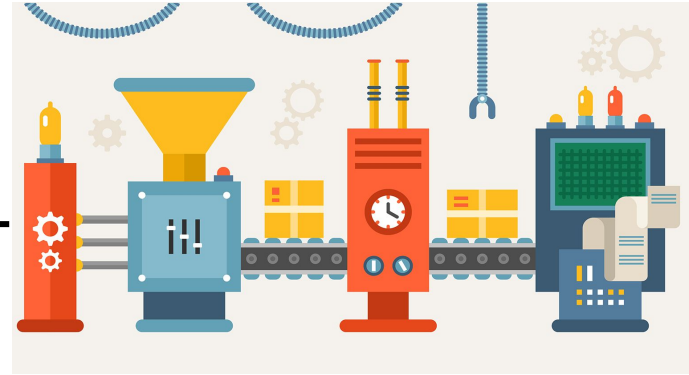
RESEARCH



DONE!!!

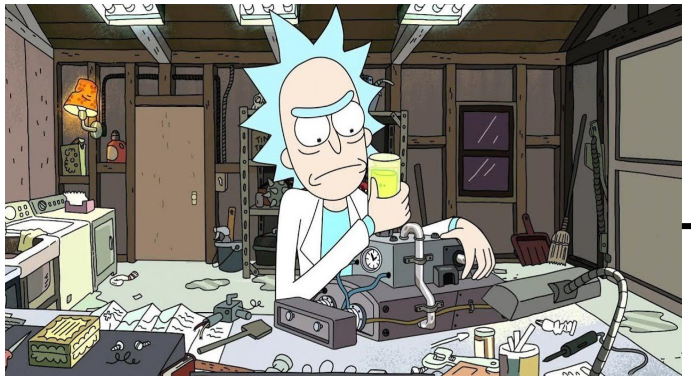


production



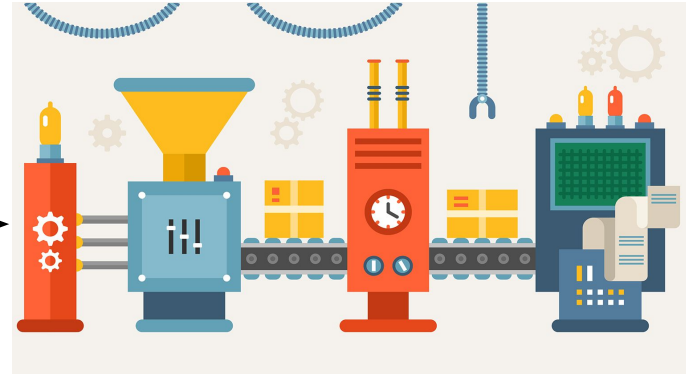
Our In-House Solution - Chronicle

RESEARCH



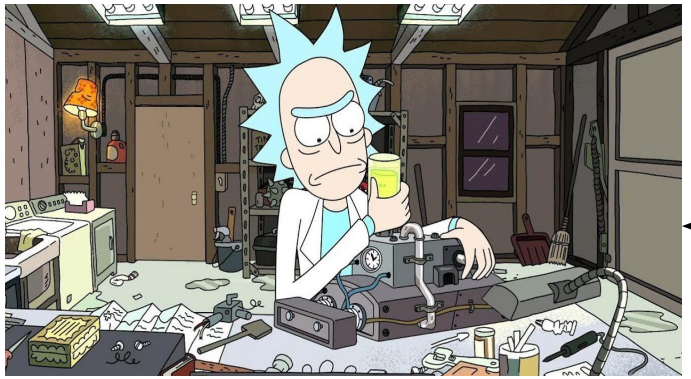
Nahhhhh... it's not good enough.
Let's add these features and these transformations

production

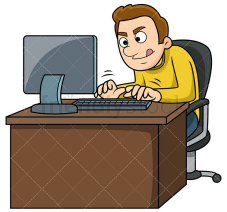


Our In-House Solution - Chronicle

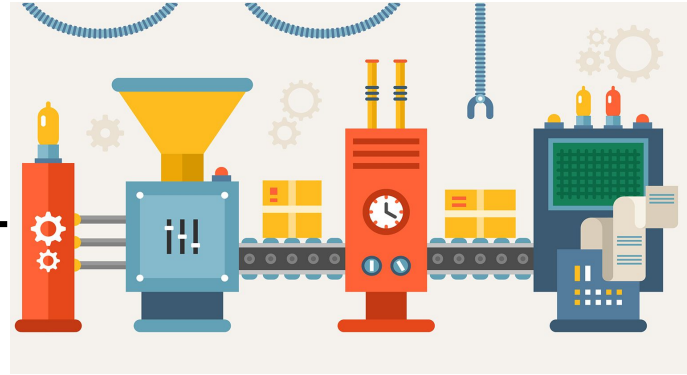
RESEARCH



DONE!!!

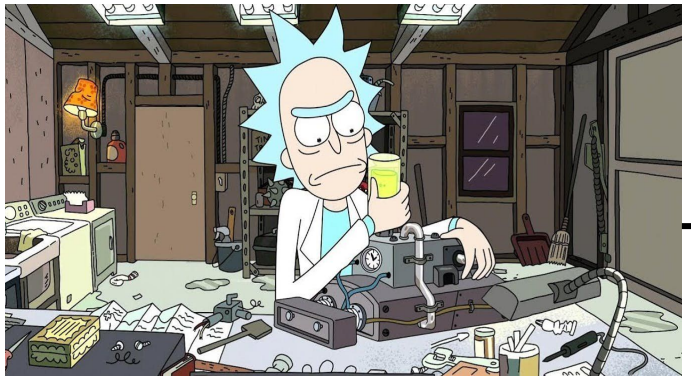


production



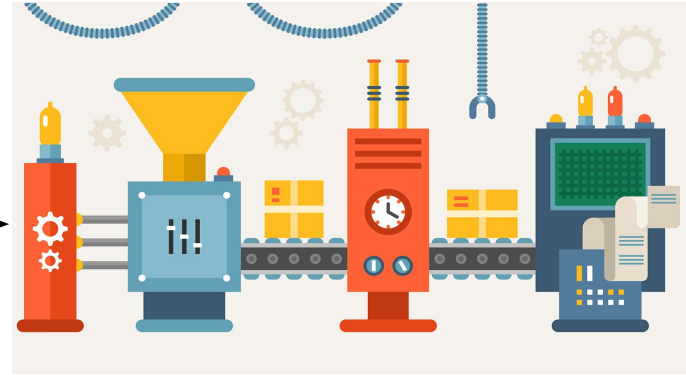
Our In-House Solution - Chronicle

RESEARCH



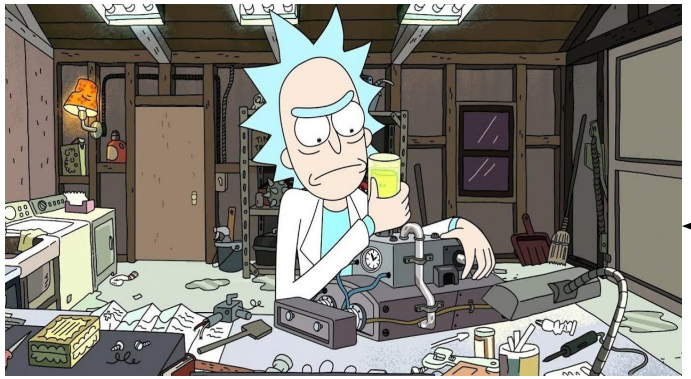
Nahhhhh... it's not good enough.
Let's add these features and these transformations

production



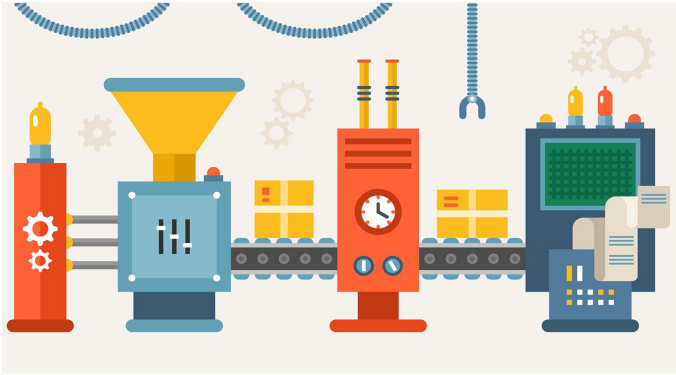
Our In-House Solution - Chronicle

RESEARCH



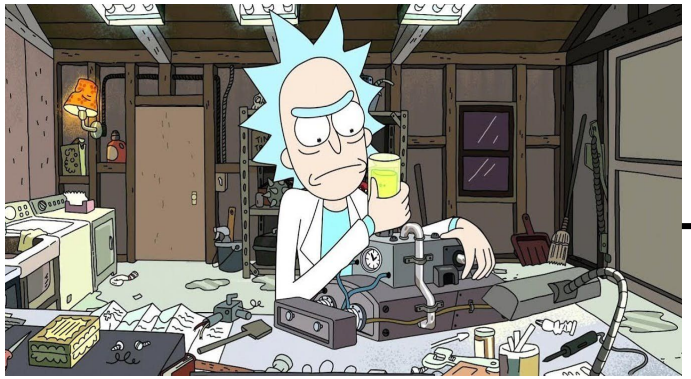
Take this scripting language called LUA
and do whatever you like with those features

production



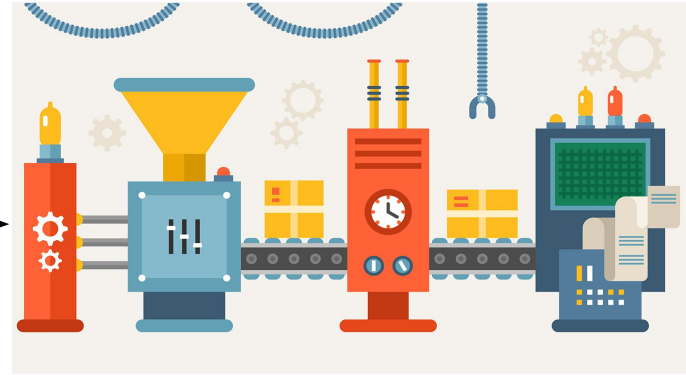
Our In-House Solution - Chronicle

RESEARCH



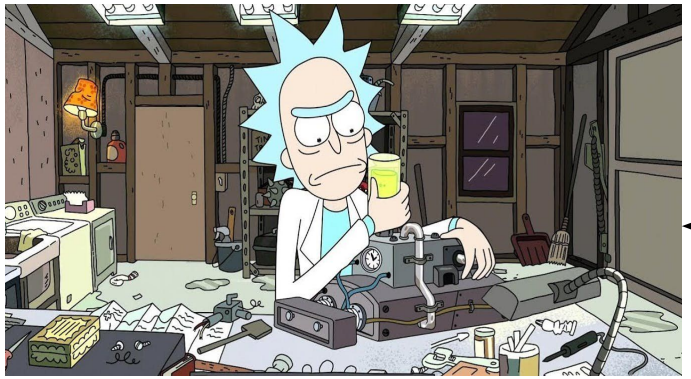
😮 ooh... Christmas came early
We'll implement them in LUA and run it.

production



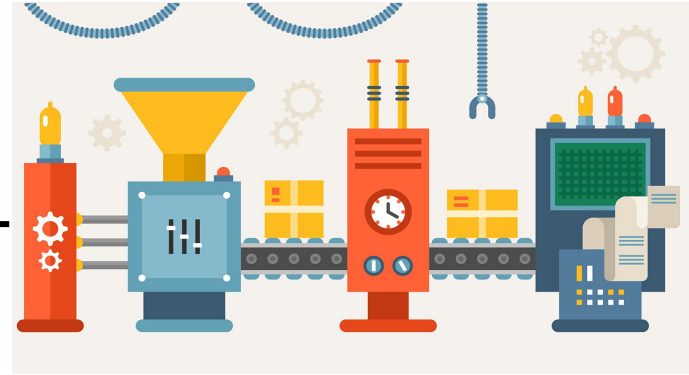
Our In-House Solution - Chronicle

RESEARCH



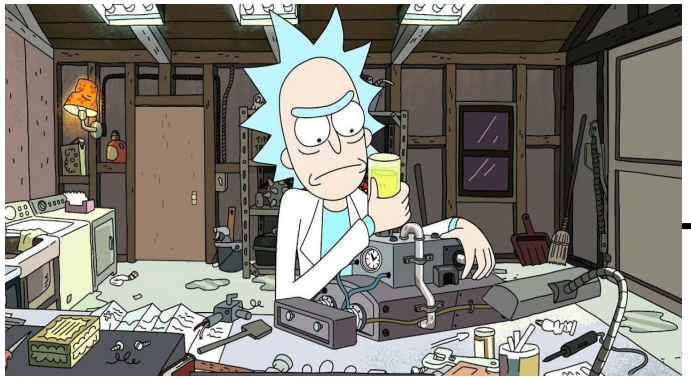
So... how is your model?!

production



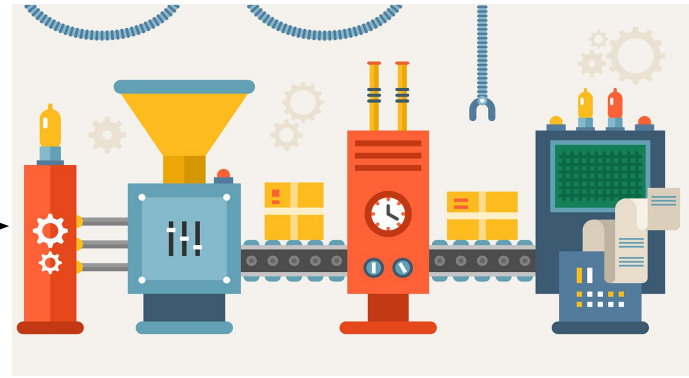
Our In-House Solution - Chronicle

RESEARCH



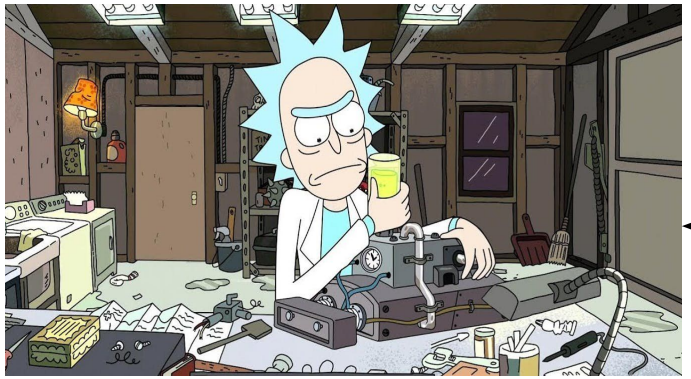
Not bad… But we will be much better with *lightGBM* instead of *XGBoost*

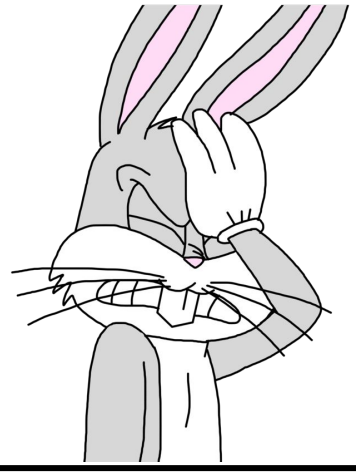
production



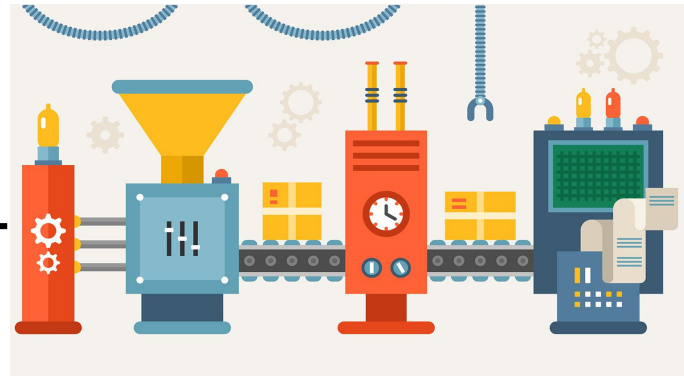
Our In-House Solution - Chronicle

RESEARCH

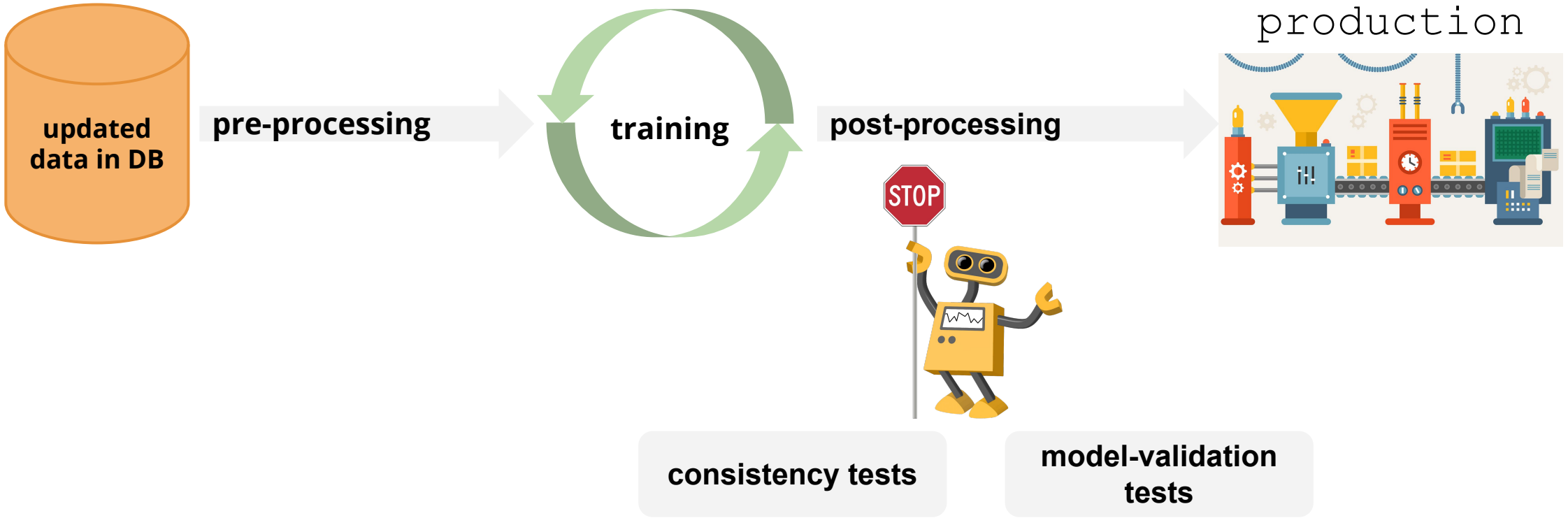




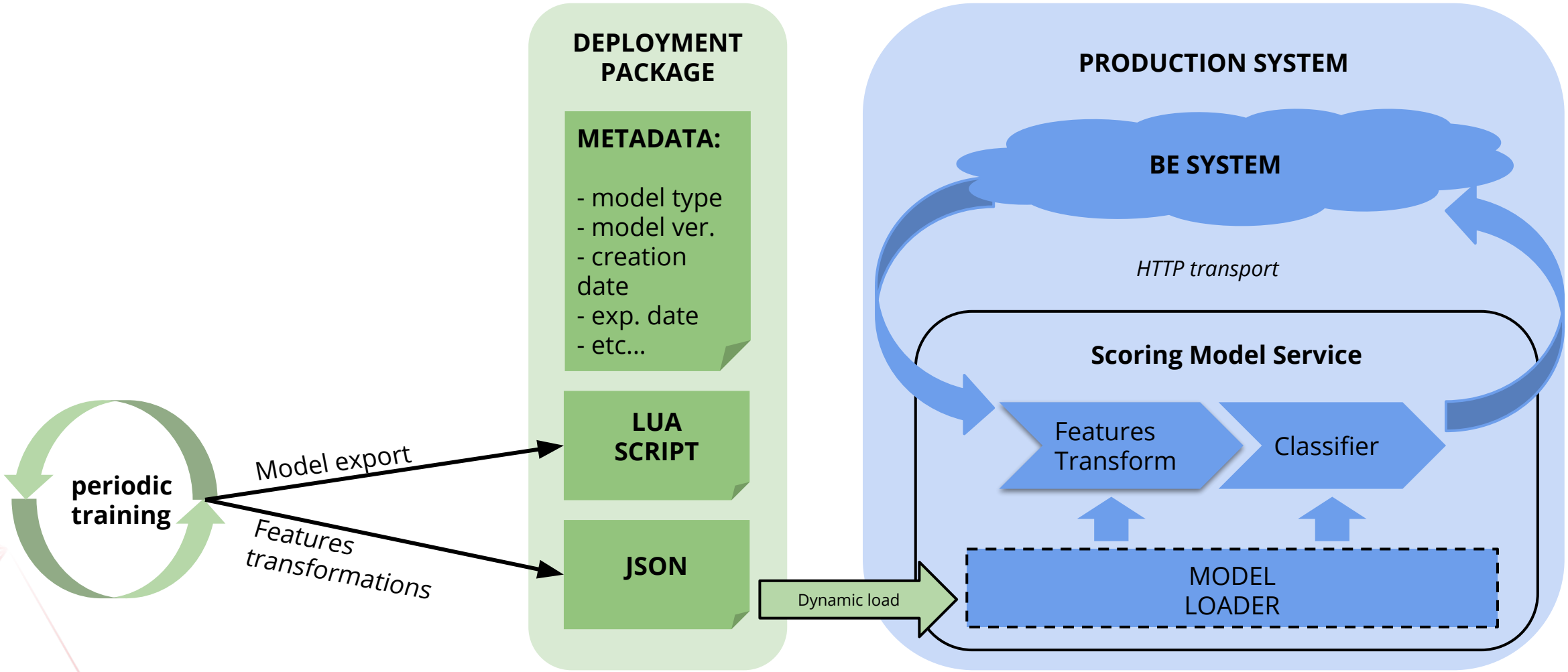
production



Our In-House Solution - periodic training



Our In-House Solution - Serve



perimeterx

Pros & Cons

- Implemented in *Go*, using known toolset and environment
- Very convenient and flexible features trans. scripting
- Easily adaptable to other boosting algorithms
- Scalable and good performance, small footprint

- Over time and iterations replicated other systems func.
- Limited only to specific boosting algorithms
- Less tested and stable than community solutions

Take Away

- Build your own or not
  - Integrated into your infrastructure (metrics / logging / scale)
  - Dynamic transformations
  - Specific decisions for edge-cases (non-model)
  - Time to market
  - Support multiple models / ML platform
- Consistency and validation tests
- Resources

Questions?