Analyze

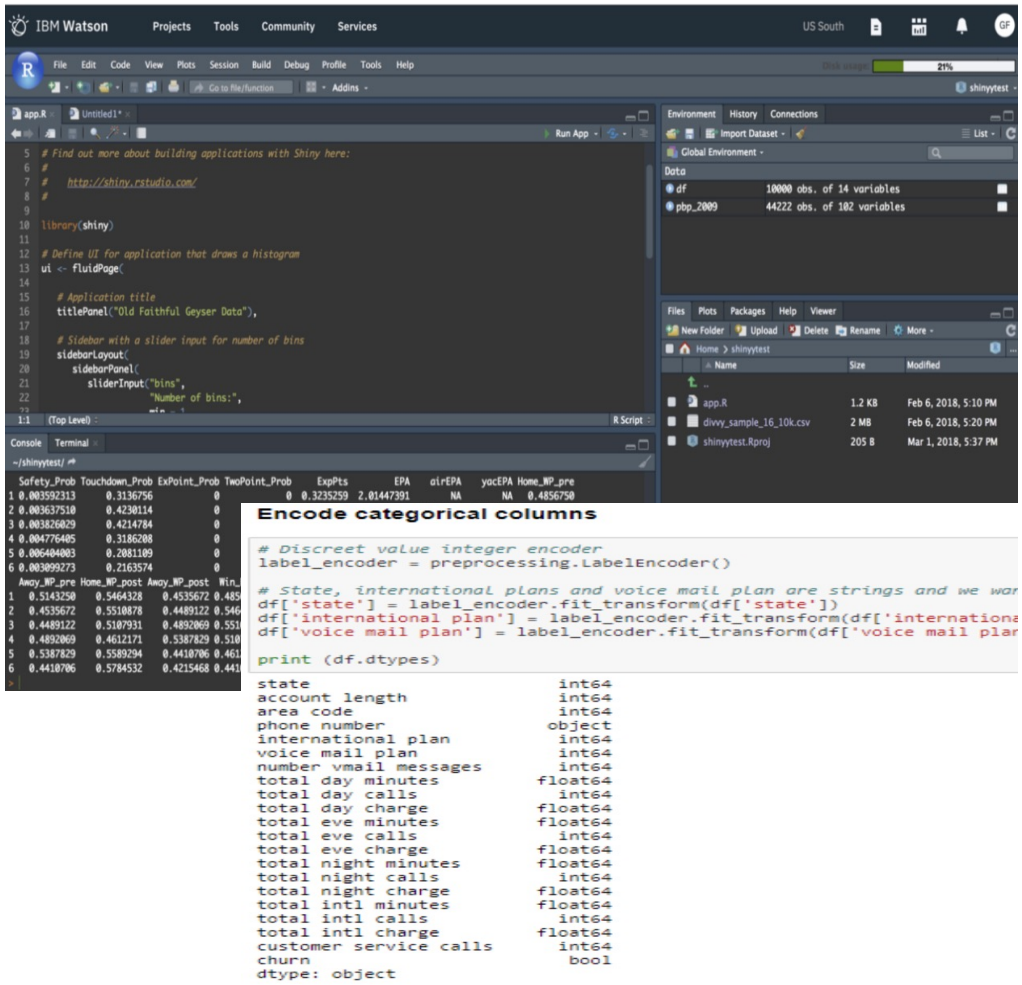
Agenda

- Identify how you can analyze data in IBM Cloud Pak for Data
- Analyze data using notebooks Identify other tools that you can use to analyze data
- Automatically analyze your data using the AutoAI tool
- Deploy machine learning (ML) models

Programmatic Manipulation

- Jupyter Notebooks and RStudio
  - **Python**, Scala and R

- Use / Install open source libraries
  - Data structure/manipulation: **Pandas**, Numpy
  - Visualizations: **Matplotlib** (most popular),Brunel, seaborn, bokeh. Plotly, ggplot, pygal, **PixieDust**, etc
  - Algorithmic: SciPy (integrals, cals), Scikit-learn, Statsmodels, etc



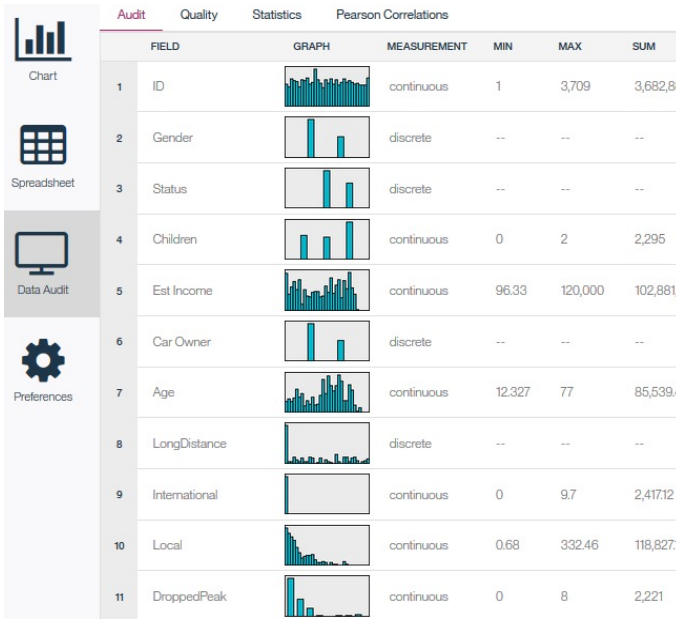
SPSS Modeler

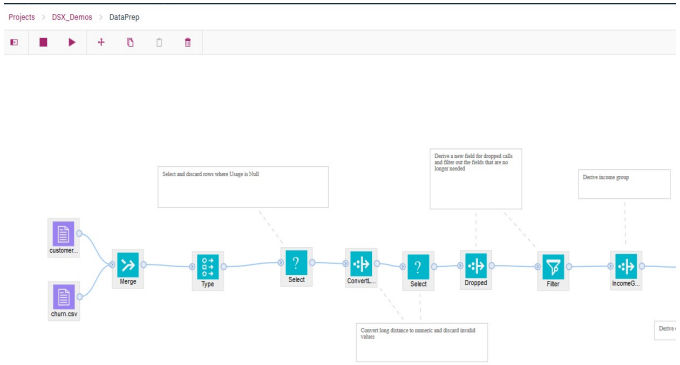
Cleansing: fix or remove data that is incorrect, incomplete, improperly formatted, or duplicated

Shaping: customize data by filtering, sorting, combining, or removing columns, and performing operations

Visualization, statistics and processing

Original data set is not modified, changes pushed to output file or data source





Auto Data Prep node

Preparing data for analysis is one of the most important steps analyzing your data and identifying fixes, screening out fields t intelligent screening techniques. You can use the algorithm in before they are made and accept or reject them as you want.

Type node

Field properties can be specified in a source node or in a sepa

Filter node

You can rename or exclude fields at any point in a stream. For level data); therefore, you can filter out the K (potassium) fielc regardless of which node it is accessed from.

Derive node

One of the most powerful features in Watson Studio is the abil several derivations, such as extracting a customer ID from a s transformations can be performed, using a variety of field ope

Filler node

Filler nodes are used to replace field values and change storag choose to replace all blanks or null values with a specific value

Reclassify node

The Reclassify node enables the transformation from one set

Binning node

The Binning node enables you to automatically create new no continuous income field into a new categorical field containing order to preserve the strength of the original association betw

RFM Analysis node

You can use the Recency, Frequency, Monetary (RFM) Analysi from you (recency), how often they purchased (frequency), an

Ensemble node

The Ensemble node combines two or more model nuggets to models, limitations in individual models may be avoided, resu models and often better.

Partition node

Partition nodes are used to generate a partition field that split to generate the model and a separate sample to test it, you ca

Set to Flag node

Use the Set to Flag node to derive flag fields based on the cate

Restructure node

The Restructure node can be used to generate multiple fields (0 and 1). The functionality of this node is similar to that of the values from another field. You can then perform aggregation c convenient if you are creating flag fields.)

Transpose node

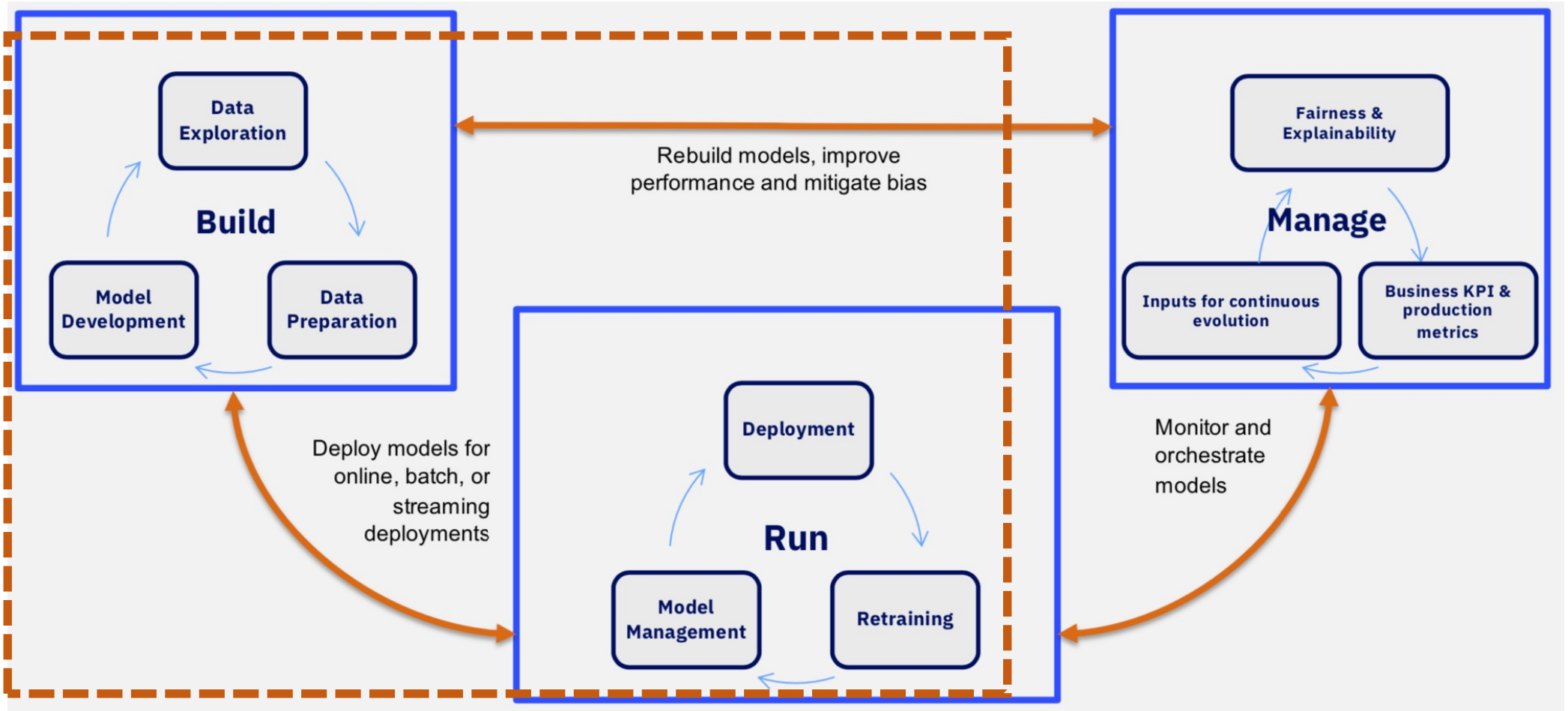
By default, columns are fields and rows are records or observa records become fields.

Field Reorder node

With the Field Reorder node, you can define the natural order Chooser.

History node

Data & AI Workflow



Machine Learning (ML)
Overview

What is Machine Learning ?

- *"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E." -- Tom Mitchell, Carnegie Mellon University*

- Learn from data to make predictions
- Computers apply statistical learning techniques to identify patterns in data and make predictions.

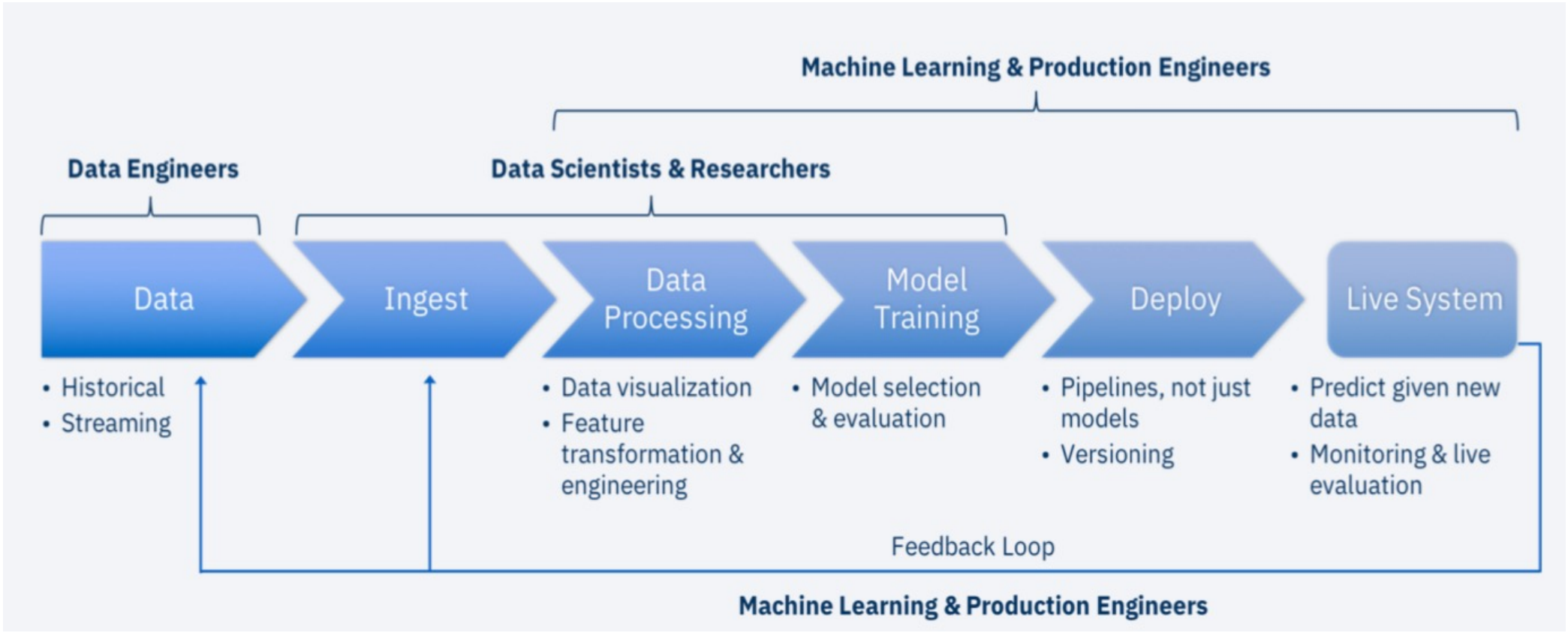
Learn from historical data to make predictions about the future

- Use example data or past experience to solve a given problem.

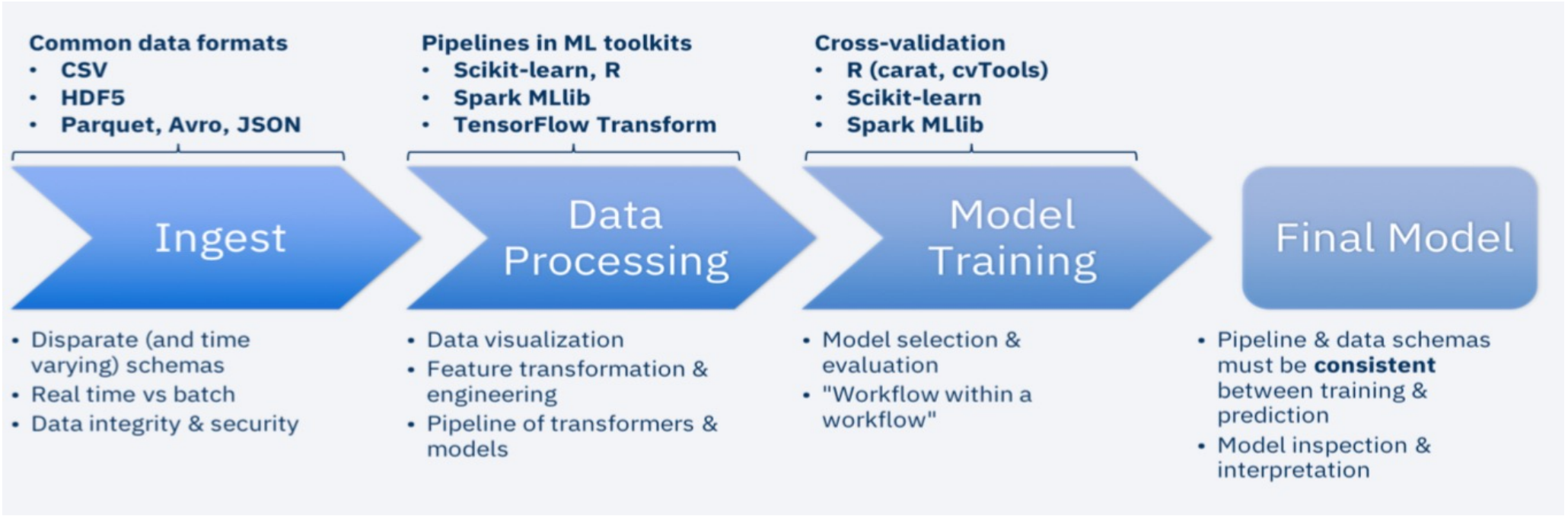
Applied Machine Learning seeks to learn from historical data to make predictions about the future, in order to make decisions

Credit: Nick Pentreath Principal Engineer, IBM Center for Open-Source Data & AI Technologies

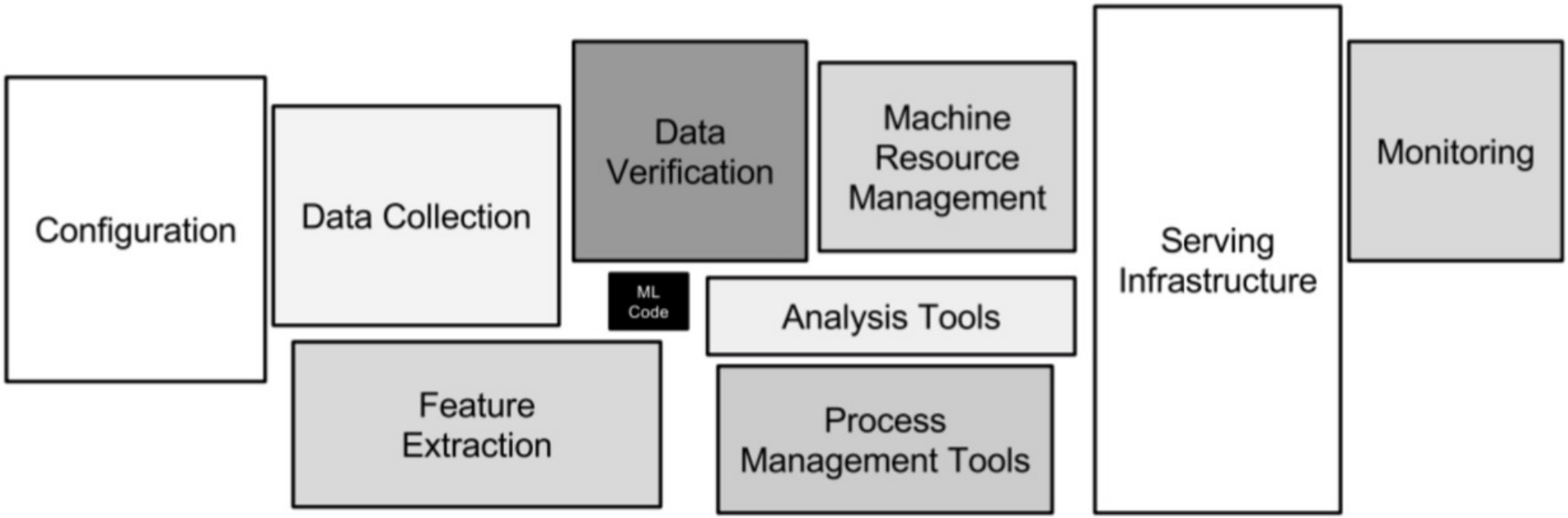
Machine Learning Across Teams



… tools


Common data formats
• CSV
• HDF5
• Parquet, Avro, JSON
Pipelines in ML toolkits
• Scikit-learn, R
• Spark MLlib
• TensorFlow Transform
Cross-validation
• R (carat, cvTools)
• Scikit-learn
• Spark MLlib
Ingest
Data Processing
Model Training
Final Model
• Disparate (and time varying) schemas
• Real time vs batch
• Data integrity & security
• Data visualization
• Feature transformation & engineering
• Pipeline of transformers & models
• Model selection & evaluation
• "Workflow within a workflow"
• Pipeline & data schemas must be consistent between training & prediction
• Model inspection & interpretation

Just one piece of the puzzle



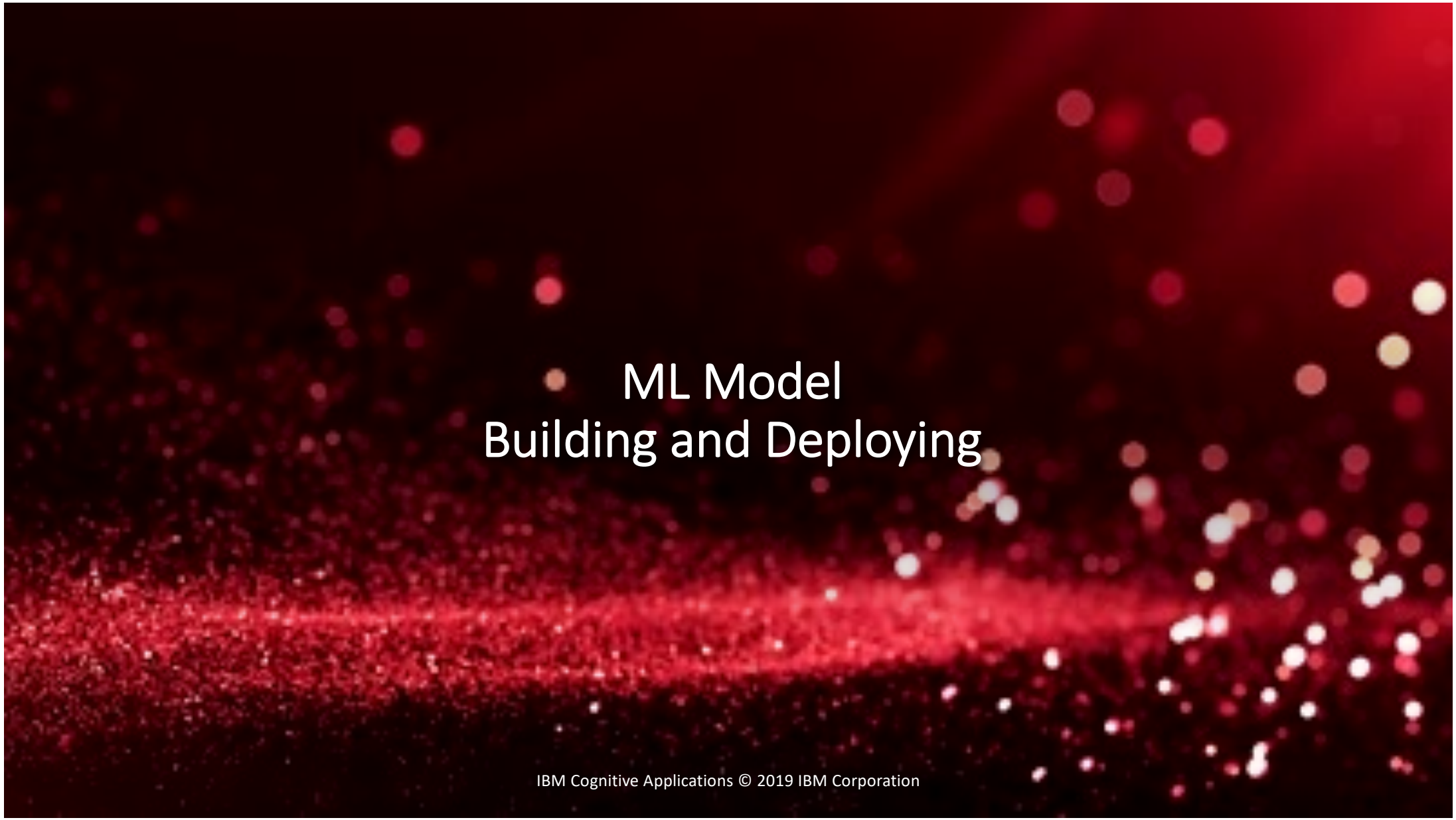
Reference

Example Linear Regression

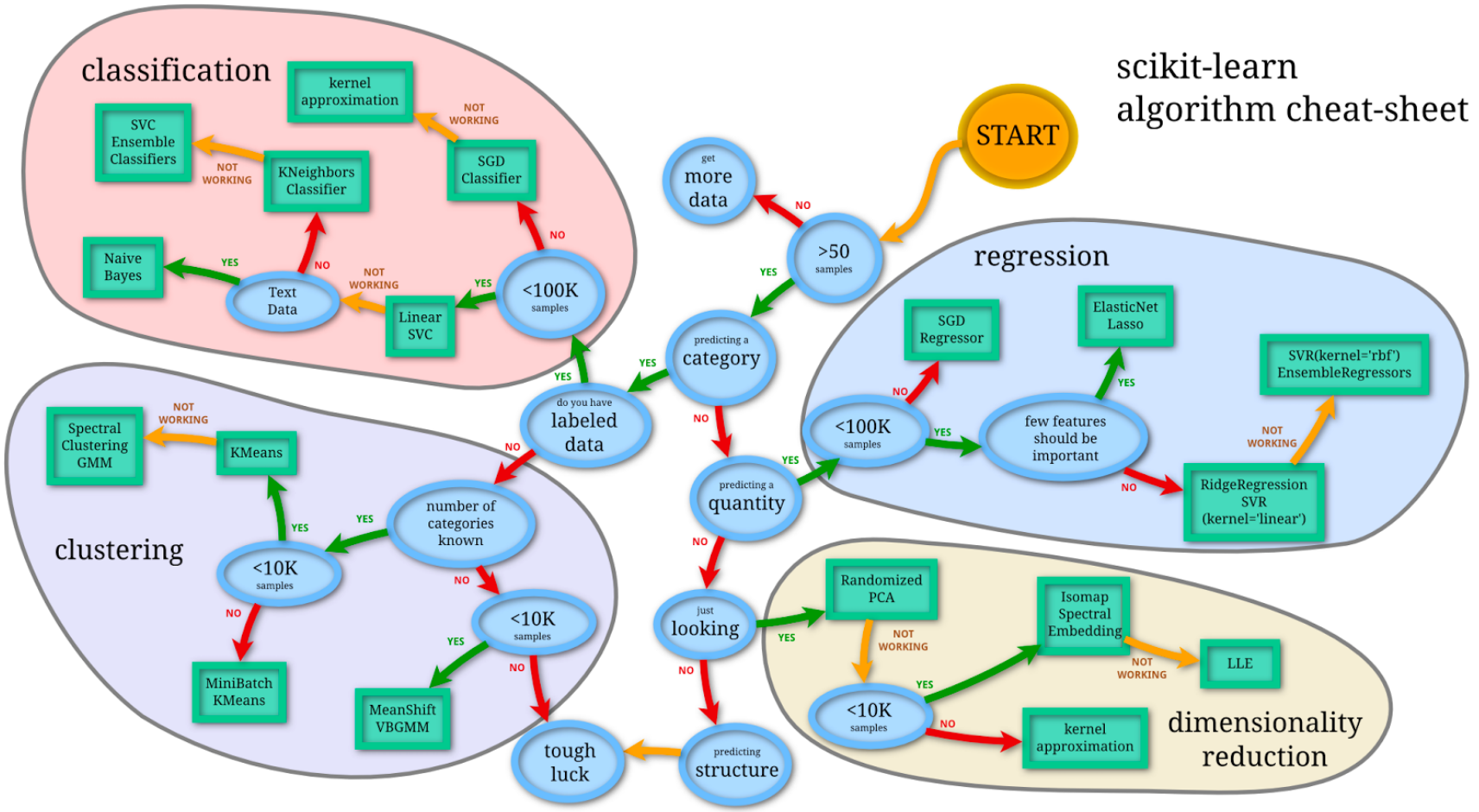
```
1  from sklearn.linear_model import LinearRegression
2  from sklearn.datasets import load_boston
3  import matplotlib.pyplot as plt
4  from sklearn.model_selection import train_test_split
5  from sklearn.metrics import mean_squared_error, r2_score
6
7  boston = load_boston()
8  X_train, X_test, y_train, y_test = train_test_split(boston.data, boston.target)
9
10 # Create a new Linear Regression Model
11 LR_model = LinearRegression()
12
13 # Train the model
14 LR_model.fit(X_train, y_train)
15
16 # store actual and predited data to draw chart
17 predicted = LR_model.predict(X_test)
18 actual = y_test
19
20
21 # The coefficients
22 print('Coefficients: \n', LR_model.coef_)
23 # The mean squared error
24 print("Mean squared error: %.2f"
25       % mean_squared_error(actual, predicted))
26 # Explained variance score: 1 is perfect prediction
27 print('Variance score: %.2f' % r2_score(actual, predicted))
```

```
# we will use WML to work with IBM Machine Learning Service
from watson_machine_learning_client import WatsonMachineLearni

# Grab your credentials from the Watson Service section in Wat
wml_credentials = {
}

# Instantiate WatsonMachineLearningAPIClient
from watson_machine_learning_client import WatsonMachineLearni
client = WatsonMachineLearningAPIClient( wml_credentials )

# store the model
published_model = client.repository.store_model(model=LR_model
                                                meta_props={'n
                                                training_data=
```

```
import json

# grab the model from IBM Cloud
published_model_uid = client.repository.get_model_uid(published_

# create a new deployment for the model
model_deployed = client.deployments.create(published_model_uid,

#get the scoring endpoint
scoring_endpoint = client.deployments.get_scoring_url(model_depl
print(scoring_endpoint)

#use the scoring endpoint to predict house median price some tes
scoring_payload = {"values": [list(X_test[0]), list(X_test[1])]}
predictions = client.deployments.score(scoring_endpoint, scoring
print(json.dumps(predictions, indent=2))
```



Andreas Mueller – ML workflow graph



Code / Notebook

- Anaconda based default or customized environments for Python (3.5) and R
  - Use Spark environment for Scala

- Don’t forget : File → Stop Kernel

Build a Machine Learning Model with Spark ML

```
In [4]: from pyspark.ml import Pipeline
        from pyspark.ml.regression import LinearRegression
        from pyspark.ml.feature import VectorAssembler

In [5]: assembler = VectorAssembler(inputCols=['SquareFeet','Bedrooms'],outputCol="features")
        lr = LinearRegression(labelCol='Price', featuresCol='features')
        pipeline = Pipeline(stages=[assembler, lr])
        model = pipeline.fit(df)
```

TensorFlow
- Version 1.5
- Version 1.11 in an Anaconda 5.2.0 environment

Spark MLlib
Spark 2.1

Caffe
Version 1.0

Predictive Model Markup Language (PMML)
Version 3.0 to 4.3

XGBoost
- XGBoost 0.6a2 and 0.71 in an Anaconda 4.2.x environment with scikit-learn 0.17 and Python 3.5
- XGBoost 0.80 in an Anaconda 5.0.1 environment with scikit-learn 0.19 and Python 3.5

scikit-learn
- scikit-learn 0.17 on Anaconda 4.2.x for Python 3.5 Runtime
- scikit-learn 0.19 on Anaconda 5.0.0 for Python 3.5 Runtime

PyTorch
Versions: 0.3, 0.4.1, 1.0

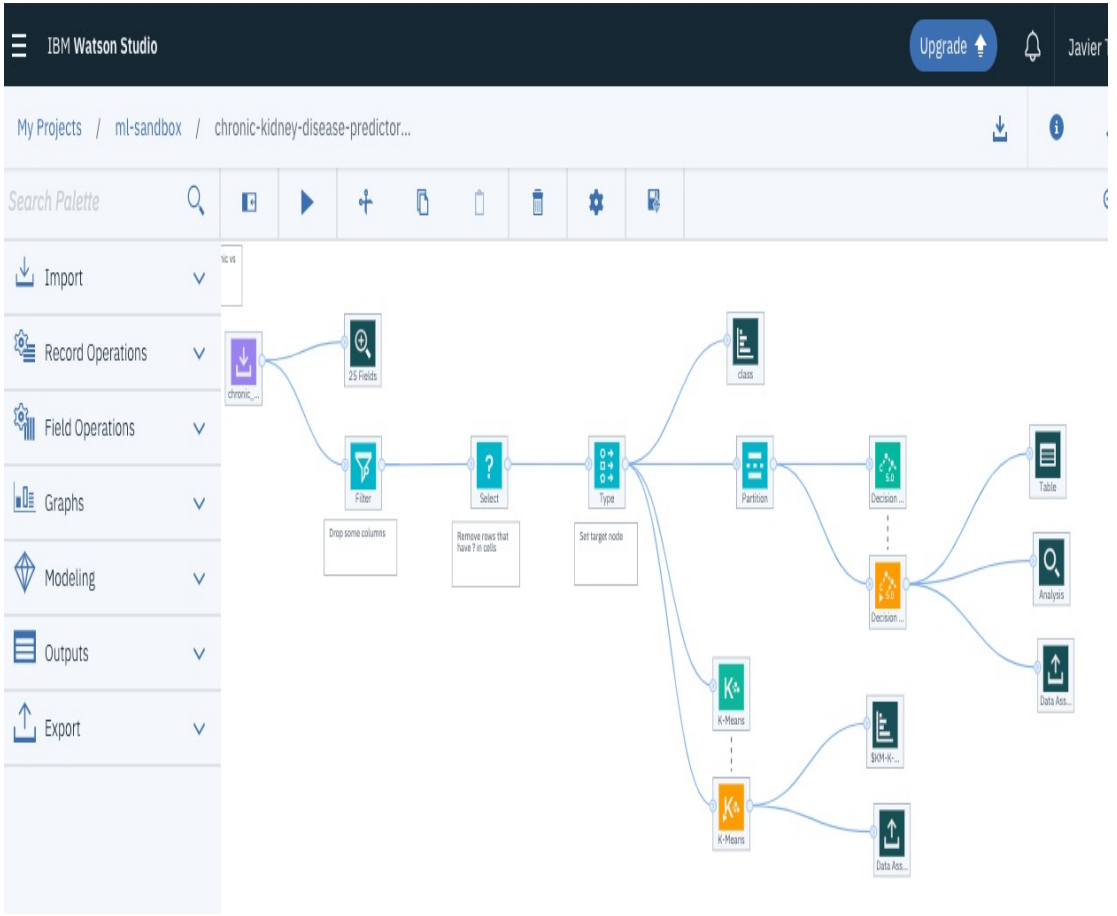
Keras
- Version 2.1.3 with Tensorflow version 1.5
- Version 2.2.4 with TensorFlow version 1.11 in an Anaconda 5.2.0 environment

IBM SPSS Modeler
- IBM SPSS Modeler 17.1
- IBM SPSS Modeler 18.0

As of June 2019

Visual Assembly

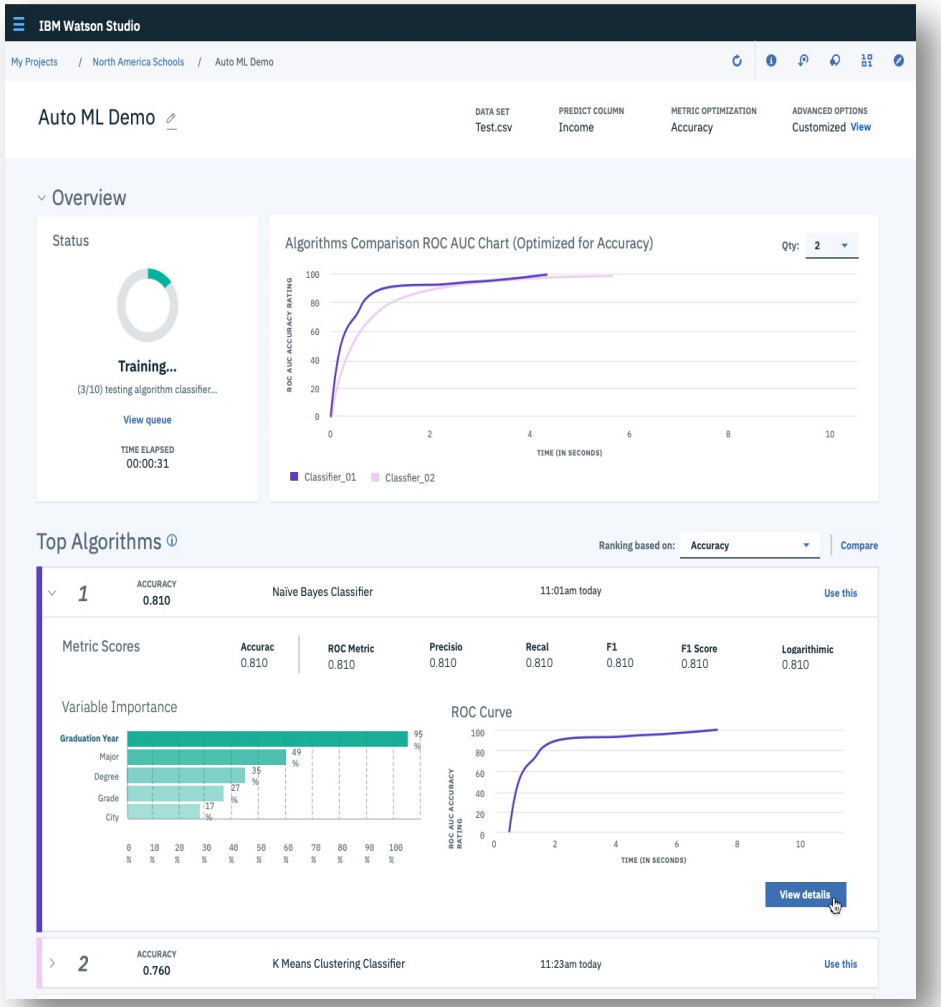
- Drag and Drop visual assembly of pipelines using either SPSS Modeler runtime or Spark.
  - Data asset nodes to bring in data
  - Data preparation nodes to modify data (or use auto data prep node)
  - Modeling nodes to develop models (decision trees, random forest, linear regression ,etc).



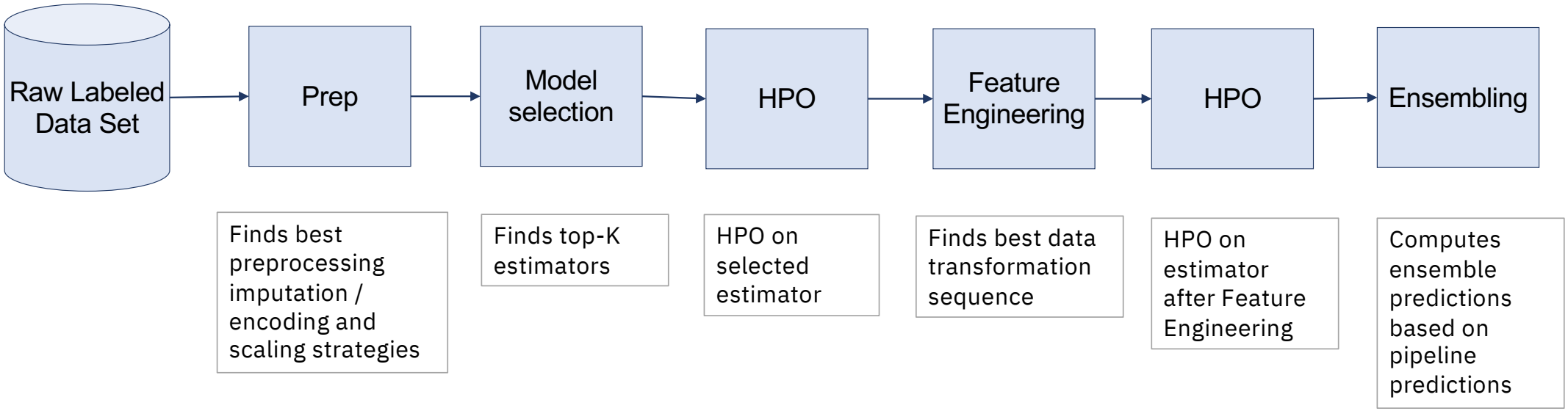
AutoAI

- Encompasses three approaches to simplify model creation:
  - Transfer knowledge learning in one deep learning system to apply to a different domain (Watson Services)
  - **AutoAI Pipeline Optimization to auto clean data, engineer features, and complete HPO to find the optimal end to end pipeline**
  - Neural Network Search to generate custom deep neural network through searching the best architectures for the input data.

- Training feedback visualizations provide real-time results to see model performance
- One-click deployment to Watson Machine Learning



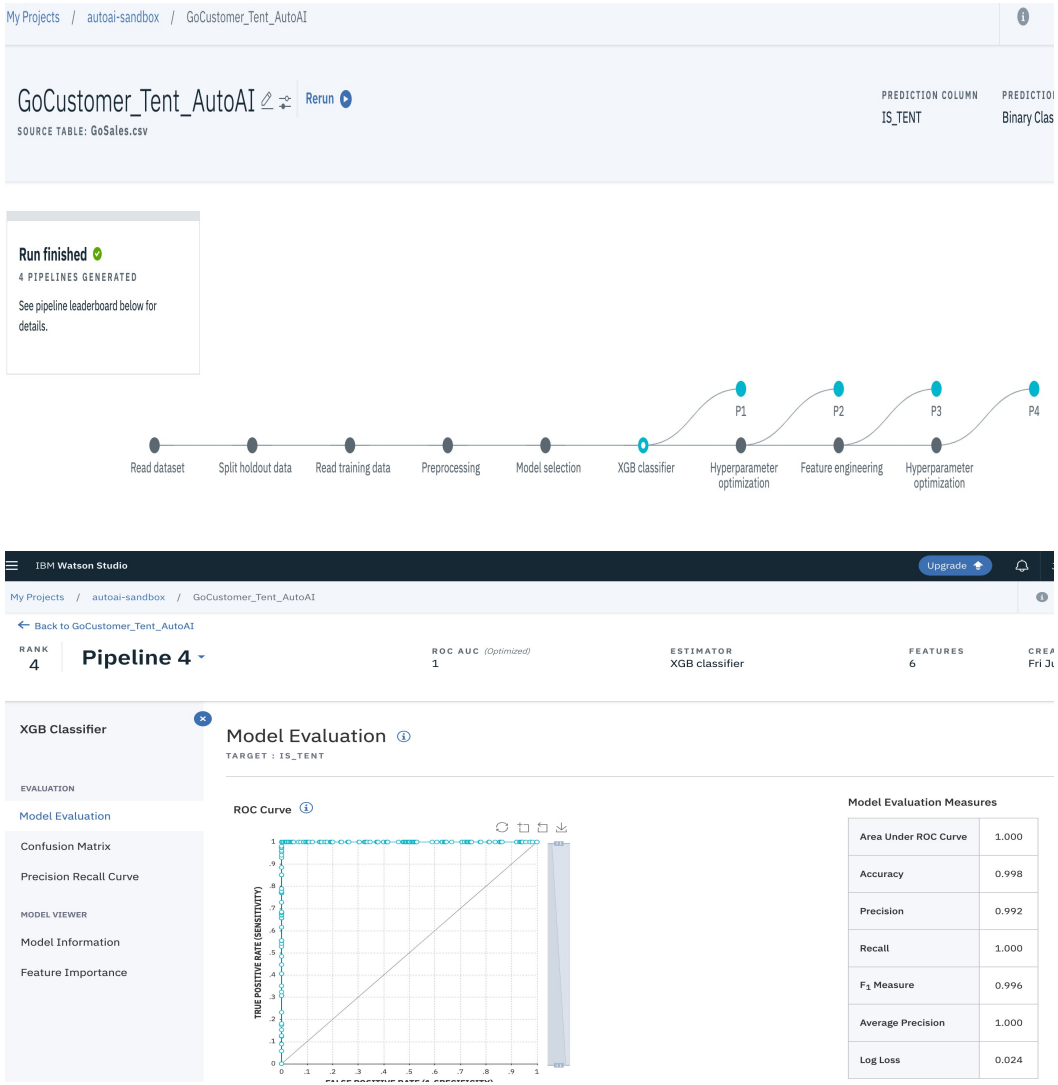
AutoAI: How does it work ?



AutoAI: Pipelines

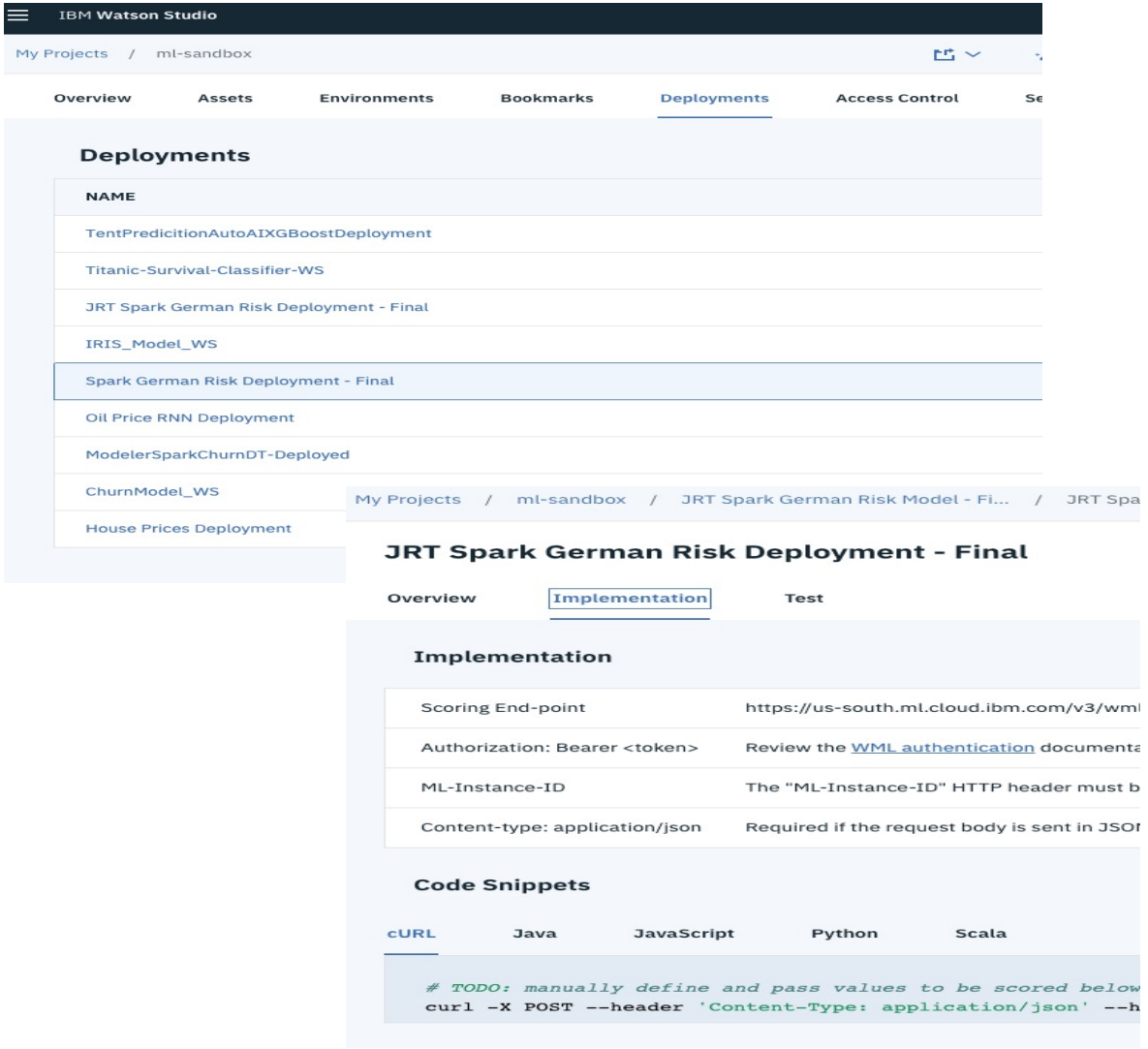
- Builds models using sklearn
- 4 pipelines per estimator
  - Baseline sklearn estimator
  - HPO optimized estimator
  - Feature engineering
  - Feature engineering + HPO

The leaderboard contains model pipelines ranked by cross-validation scores (defaults = AUC score, Accuracy, and RMSE)



Deploying Models

- Watson Machine Learning can host model (creates scoring endpoint)
- Deployments via Python Client, REST API, Watson Studio GUI
  - Online web service
  - Batch
  - CoreML
- Deploy models directly or Python functions to encapsulate model + logic
  - Preprocessing data, error handling, custom model orchestration, etc.



Thank You

- Binu Midhun
- Email: bmidhun1@in.ibm.com
- Twitter: @binu_midhun