Welcome to
KNIME Big Data Workshop

Going live at:

Berlin 5:00 PM (CEST)
New York City 11:00 AM (EDT)
Austin 10:00 AM (CDT)
London 4:00 PM (GMT)
Before we start...

- Please use the Q&A section to post your questions.
- Upvote for your favorite questions.
- Session is recorded and will be available on YouTube.
Before we start...

• Please use the Q&A section to post your questions.
• Upvote for your favourite questions.
• Session is recorded and will be available on YouTube.
Agenda

• What is "big data"?

• KNIME Big Data Extensions
  – Introduction to Hadoop and Spark
  – KNIME Big Data Connectors
  – KNIME Extension for Apache Spark
  – KNIME H2O Sparkling Water Integration
  – KNIME Workflow Executor for Apache Spark
What is "Big Data" about?

"...ways to analyze, systematically extract information from [...] data sets that are too large or complex to be dealt with by traditional data-processing application software." [1]

The three Vs of what makes data "big":
- Volume (size of data)
- Variety (tabular, text, images, video, audio, time series, ...)
- Velocity (produced fast and continuously)

Goal of big data technologies:
Enable predictive or other types of advanced analytics to extract value from big data.

KNIME Analytics Platform: Open for Every Data, Tool, and User
KNIME Big Data Extensions

• Introduction to Hadoop and Spark
• KNIME Big Data Connectors
• KNIME Extension for Apache Spark
• KNIME H2O Sparkling Water Integration
• KNIME Workflow Executor for Apache Spark
Apache Hadoop

• Open-source project for distributed storage and processing of large data sets

• Designed to scale up to thousands of machines

• First release in 2006
  – Rapid adoption, promoted to top level Apache project in 2008

• Spawned diverse ecosystem of products
Hadoop Ecosystem

- Storage: HDFS
- Resource Management: YARN
- Processing: MapReduce, Tez, Spark
- SQL: HIVE
HDFS

- Hadoop distributed file system
- Stores large files across multiple machines
Data Replication

- All blocks of a file are stored as sequence of blocks
- Blocks of a file are replicated for fault tolerance (usually 3 replicas)
  - improves data reliability, availability, and network bandwidth utilization
YARN

• Cluster resource management system

• Two elements
  – Resource manager (one per cluster):
    • Knows where worker nodes are located and how many resources they have
    • Scheduler: Decides how to allocate resources to applications
  – Node manager (many per cluster):
    • Launches application containers
    • Monitor resource usage and report to Resource Manager
MapReduce

Map applies a function to each element
*For each word emit: word, 1*

Reduce aggregates a list of values to one result
*For all equal words sum up count*
Hive

- **SQL database** on top of files in HDFS
- Provides data summarization, query, and analysis
- Interprets a set of files as a database table (schema information to be provided)
- Translates SQL queries to MapReduce, Tez, or Spark jobs
- Supports various file formats:
  - Text/CSV
  - SequenceFile
  - Avro
  - ORC
  - Parquet
Hive

SQL

select * from table

MapReduce / Tez / Spark

MAP(...) REDUCE(...)
Spark

• Cluster computing framework for large-scale data processing
• In-memory computing
  – much (!) faster than MapReduce
• Programmatic interface (Scala, Java, Python, R)
• Great for:
  – Iterative algorithms
  – Interactive data analysis
Spark – Data Representation

DataFrame:

- **Table-like**: Collection of rows, organized in columns with names and types

- **Immutable**:
  - Data manipulation = creating new DataFrame from an existing one by applying a function on it

- **Lazily evaluated**:
  - Functions are not executed until an action is triggered, that requests to actually see the row data

- **Distributed**:
  - Each row belongs to exactly one partition
  - Each partition is held by a Spark Executor

<table>
<thead>
<tr>
<th>Name</th>
<th>Surname</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>Doe</td>
<td>35</td>
</tr>
<tr>
<td>Jane</td>
<td>Roe</td>
<td>29</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Spark – Lazy Evaluation

- Functions ("transformations") on DataFrames are not executed immediately
- Spark keeps record of the transformations for each DataFrame
- The actual execution is only triggered once the data is needed
- Offers the possibility to optimize the transformation steps

![Diagram showing Spark pipeline with Triggers evaluation note]
Spark Context

- Spark Context
  - Main entry point for Spark functionality
  - Represents connection to a Spark cluster
  - Allocates resources on the cluster
KNIME Big Data Extensions

- Introduction to Hadoop and Spark
- KNIME Big Data Connectors
- KNIME Extension for Apache Spark
- KNIME H2O Sparkling Water Integration
- KNIME Workflow Executor for Apache Spark
Database Extension

- Visually assemble complex SQL statements (no SQL coding needed)
- Connect to all JDBC-compliant databases
- Harness the power of your database within KNIME
Database Connectors

- Many dedicated DB Connector nodes available
- If connector node missing, use DB Connector node with JDBC driver
In-Database Processing

- Database Manipulation nodes generate SQL query on top of the input SQL query (brown square port)
- SQL operations are executed on the database!
Export Data

- Writing data back into database
- Exporting data into KNIME
KNIME Big Data Connectors

- Built upon Database extension
- Include drivers/libraries for HDFS, Hive, Impala and Databricks
- Preconfigured connectors
  - Hive
  - Impala
  - Databricks (Thriftserver)
Hive Connector

• Creates JDBC connection to Hive
• On unsecured clusters no password required
Preferences: Registering proprietary JDBC drivers

Useful for:
- Cloudera Hive/Impala JDBC drivers
- Databricks JDBC Driver
Loading Data into Hive/Impala

- Connectors are from KNIME Big Data Connectors Extension
- Use DB Table Creator and DB Loader from regular DB framework
Demo: 01_Flight_Delay_Statistics_Impala

Shortened URL to KNIME Hub folder: https://tinyurl.com/ycyh3kya
KNIME Big Data Extensions

• Introduction to Hadoop and Spark
• KNIME Big Data Connectors
• KNIME Extension for Apache Spark
• KNIME H2O Sparkling Water Integration
• KNIME Workflow Executor for Apache Spark
KNIME Extension for Apache Spark

- Based on Spark MLlib
- Scalable machine learning library
- Supports algorithms for
  - Classification (decision tree, naïve bayes, ...)
  - Regression (logistic regression, linear regression, ...)
  - Clustering (k-means)
  - Collaborative filtering (ALS)
  - Dimensionality reduction (SVD, PCA)
Spark Contexts: Creating

Three nodes to create a Spark context:

• Create Local Big Data Environment
  – Runs Spark locally on your machine (no cluster required)
  – Good for workflow prototyping

• Create Spark Context (Livy)
  – Requires a cluster that provides the Livy service
  – Good for production use

• Create Databricks Environment
  – Requires a Databricks cluster on AWS or Azure
  – Provides DB connection, DBFS and Spark
Create Spark Context (Livy)

- Allows to use Spark nodes on clusters with Apache Livy
- Compatible with CDH, HDP, HDInsight and EMR

Also supported:

- Secure WebHDFS via KNOX Connection
- WebHDFS Connection
- HDFS Connection
- Amazon S3 Connection
- HttpFS Connection
- Create Spark Context (Livy)
Import Data to Spark

• From KNIME

• From CSV file in HDFS

• From Hive

• From other sources

• From Database
Modularize and Execute Your Own Spark Code: PySpark Script
MLlib Integration: Familiar Usage Model

- Usage model and dialogs similar to existing nodes
- No coding required
- Various algorithms for classification, regression and clustering supported
MLlib Integration: Spark MLlib Model Port

- MLlib model ports for model transfer
- Model ports provide more information about the model itself
Demo: 02_Taxi_Demand_Prediction_Training_workflow

Shortened URL to KNIME Hub folder: https://tinyurl.com/ycyh3kya
KNIME Big Data Extensions

- Introduction to Hadoop and Spark
- KNIME Big Data Connectors
- KNIME Extension for Apache Spark
- KNIME H2O Sparkling Water Integration
- KNIME Workflow Executor for Apache Spark
H2O Integration

- KNIME integrates the H2O machine learning library
- H2O: Open source, focus on scalability and performance
- Supports many different models
  - Generalized Linear Model
  - Gradient Boosting Machine
  - Random Forest
  - k-Means, PCA, Naive Bayes, etc.
- Includes support for MOJO model objects for deployment
- Sparkling water = H2O on Spark
The H2O Sparkling Water Integration

Train Churn Prediction Model on Spark using H2O Sparkling Water

Create Local Big Data Environment
Create H2O Sparkling Water Context
Database Table Selector
Hive to Spark
Spark to H2O
H2O Partitioning
H2O Gradient Boosting Machine Learner
H2O Model to MOJO
H2O MOJO Writer

Select * FROM customer_data

70% Training 30% Testing

Deploy Mojo Model on KNIME Server

H2O MOJO Reader
Read Churn Prediction Model
H2O MOJO Predictor (Classification)
Container Input (JSON)
JSON to Table
Container Output (JSON)

Input from REST call
Node 265
Result of REST call
KNIME Big Data Extensions

- Introduction to Hadoop and Spark
- KNIME Big Data Connectors
- KNIME Extension for Apache Spark
- KNIME H2O Sparkling Water Integration
- KNIME Workflow Executor for Apache Spark
KNIME Workflow Executor for Apache Spark
Use Cases & Limitations

• Each workflow replica processes the rows of one partition!

• **Good match for:**
  – KNIME nodes that operate row-by-row
    • Many pre- and postprocessing nodes
    • Predictor nodes
    • Nodes that are streamable
  – Parallel execution of standard KNIME workflows on “small” data
    • Hyper-parameter optimization

• **Bad match for:** Any node that needs all rows, such as
  – GroupBy, Joiner, Pivoting, ...
  – Model learner nodes