



An Online Spark Pipeline: Semi-Supervised Learning and Online Retraining with Spark Streaming.

Presented: J. White Bear

IBM Spark Technology Center



2

- Founded in 2015.
- Location:
 - Physical: 505 Howard St., San Francisco CA
 - Web: <http://spark.tc> Twitter: [@apachespark_tc](https://twitter.com/apachespark_tc)
- Mission:
 - **Contribute** intellectual and technical capital to the Apache Spark community.
 - Make the core technology **enterprise- and cloud-ready**.
 - Build **data science skills** to drive intelligence into business applications — <http://bigdatauniversity.com>
- Key statistics:
 - About 50 developers, co-located with 25 IBM designers.
 - Major contributions to Apache Spark <http://jiras.spark.tc>
 - Apache SystemML is now a top level Apache project !
 - Founding member of UC Berkeley AMPLab and RISE Lab
 - Member of R Consortium and Scala Center

About Me

Education

- **University of Michigan- Computer Science**
 - **Databases, Machine Learning/ Computational Biology, Cryptography**
- **University of California San Francisco, University of California Berkeley-**
 - **Multi-objective Optimization/ Computational Biology/ Bioinformatics**
- **McGill University**
 - **Machine Learning/ Multi-objective Optimization for Path Planning/ Cryptography**

Industry

- **IBM**
- **Amazon**
- **TeraGrid**
- **Pfizer**
- **Research at UC Berkeley, Purdue University, and every university I ever attended. 😊**

Fun Facts (?)

I love research for its own sake. I like robots, helping to cure diseases, advocating for social change and reform, and breaking encryptions. Also, most activities involving the Ocean and I usually hate taking pictures. 😊



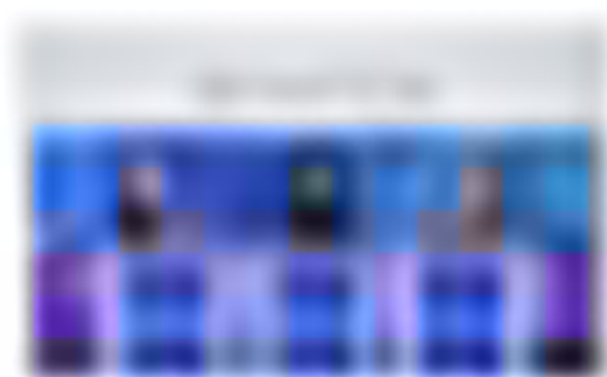
Why do we need online semisupervised learning?



Stock Prediction



Facial Classification



Text/Image Classification



Image Classification



Why online learning?

- Incremental/Sequential learning for real-time use cases
- Predicts/learns from the newest data
- Optimized for low latency in real-time cases
- Often used in conjunction with streaming data

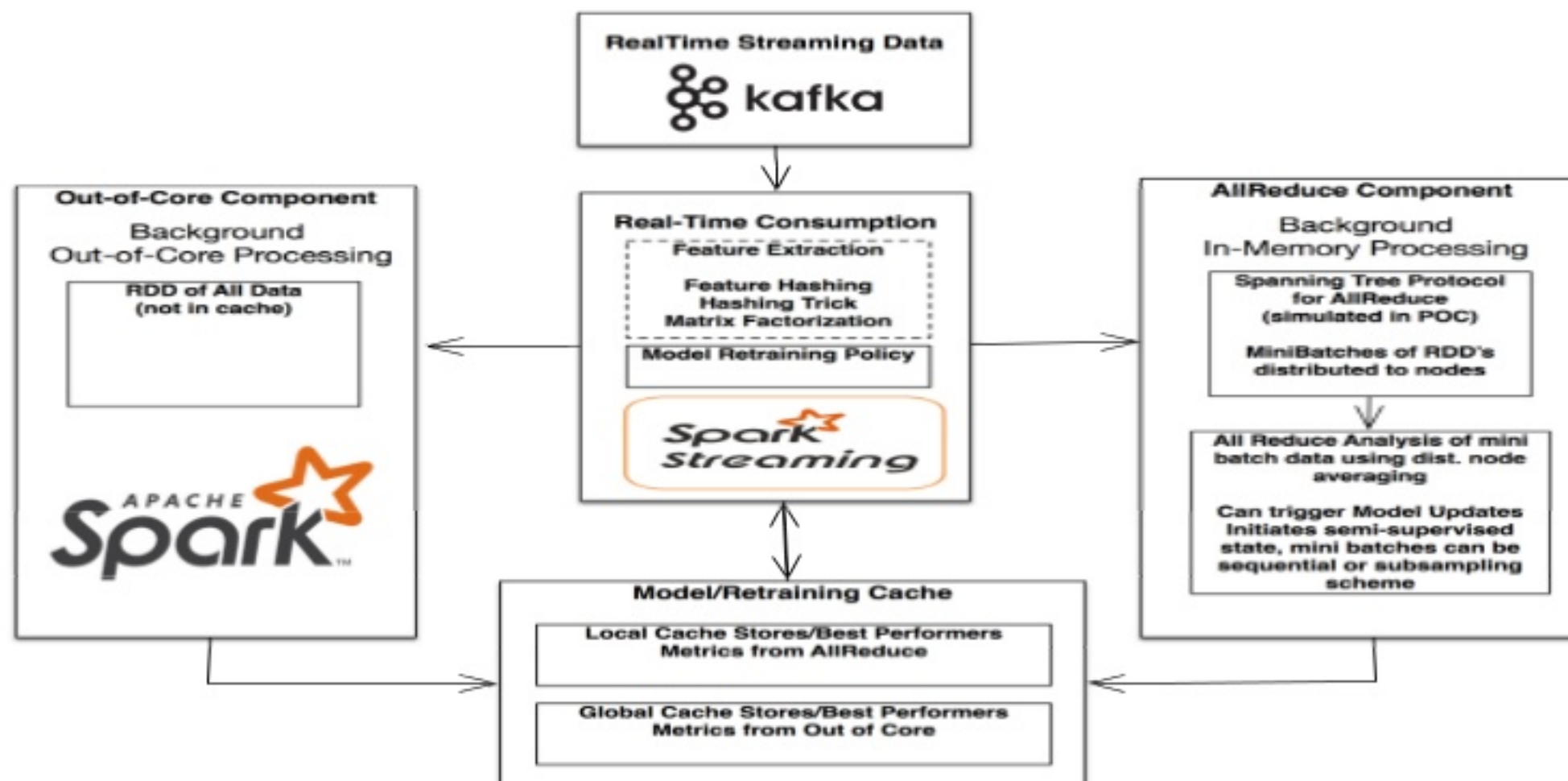
Semi-supervised learning

- Smaller training sets, less labeled data
- Classifying or predicting unlabeled data
- The underlying distribution $P(x,y)$ may or may not be known/ stable (PAC learning)
- How/Why do we bring these ideas together?

Key Challenges

- Acquiring a sufficient ratio training data
 - Batch learning is much slower and more difficult to update a model
- Maintaining accuracy
 - Concept drift
 - Catastrophic events
- Meeting latency requirements particularly in real-time scenarios, like autonomous vehicles
- What about unlabeled data?....

The Framework: Hybrid Online and Semi-Supervised learning



Why this framework?

- Abundance of unlabeled data requires semi-supervised learning
 - Real-time learning in the IoT setting
- Semi-Supervised learning can improve predictions
 - Empirically studied for online use case with Big Data
- We need to address the challenges of incremental sequential learning without losing valuable historical data
- We need to optimize for low latency without overly sacrificing accuracy
- You may have an online case but you are not guaranteed labeled data and definitely not in a timely fashion
- Hybrid frameworks address these challenges and allow for future data mining to understand and correct for how your data changes over time

Online and Semi-supervised Learning: Online Constraint

$$\min_{\mathbf{w} \in \mathcal{R}^d} \sum_{i=1}^n l(\mathbf{w}^\top \mathbf{x}_i; y_i) + \lambda \mathcal{R}(\mathbf{w})$$

Supervised learning over a batch



$$\frac{1}{n} \sum_{i=1}^n l(\mathbf{w}^\top \mathbf{x}_i; y_i) + \frac{\lambda}{n} \mathcal{R}(\mathbf{w})$$

Online learning over a mini-batch from a streaming data set



$$\frac{m}{n} \sum_{i \in \mathbb{S}_k}^n l(\mathbf{w}^\top \mathbf{x}_i; y_i) + \frac{\lambda}{n} \mathcal{R}(\mathbf{w})$$

Distributed learning over a mini-batch from a streaming data set

Online and Semi-supervised Learning: Semi-supervised Constraint

Let x_p be an example where $x_p = (x_{p1}, x_{p2}, x_{p2}, \dots, x_{pD}, \omega)$ where x_p belongs to class ω and a \mathcal{D} dimensional space. x_{pi} is the value of the i th feature of the p th sample.

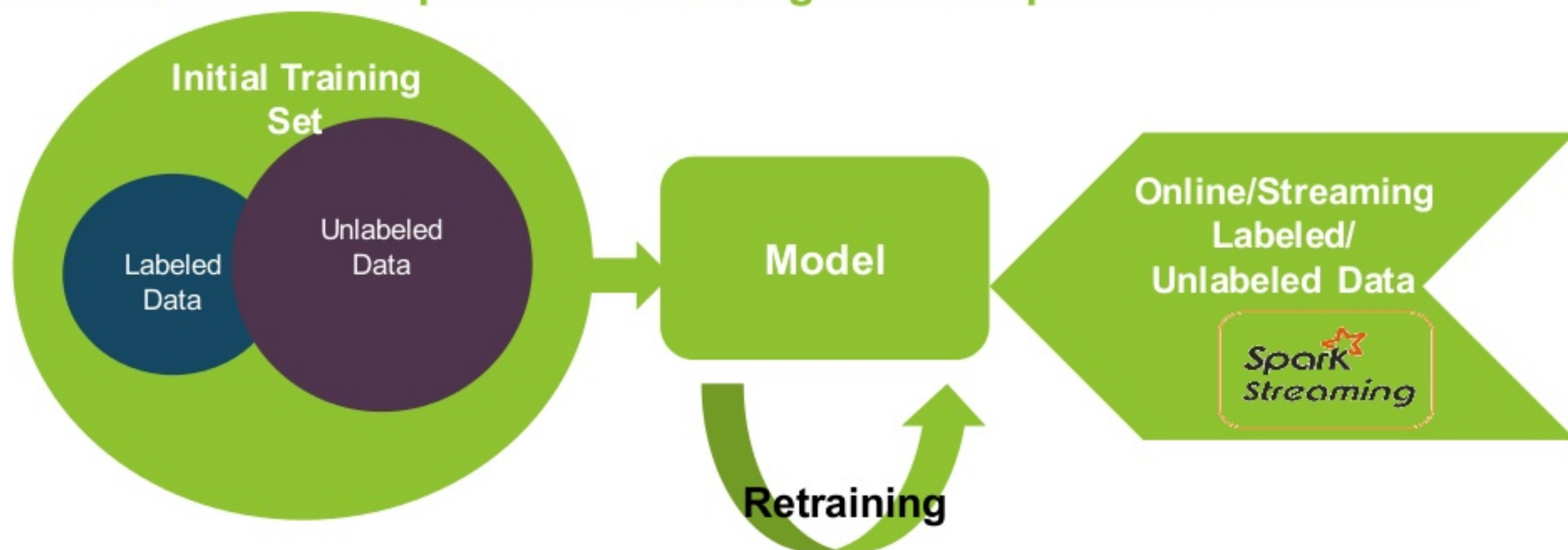
Assume a labeled set. \mathcal{L} with n instances of x , with ω known. Assume a labeled set. \mathcal{U} with m instances of x , with ω unknown. Assume that the number of labeled instances \mathcal{L} , is less than the number of unlabeled instances, \mathcal{U} .

$\mathcal{L} \cup \mathcal{U}$ represents the training set, \mathcal{T} .

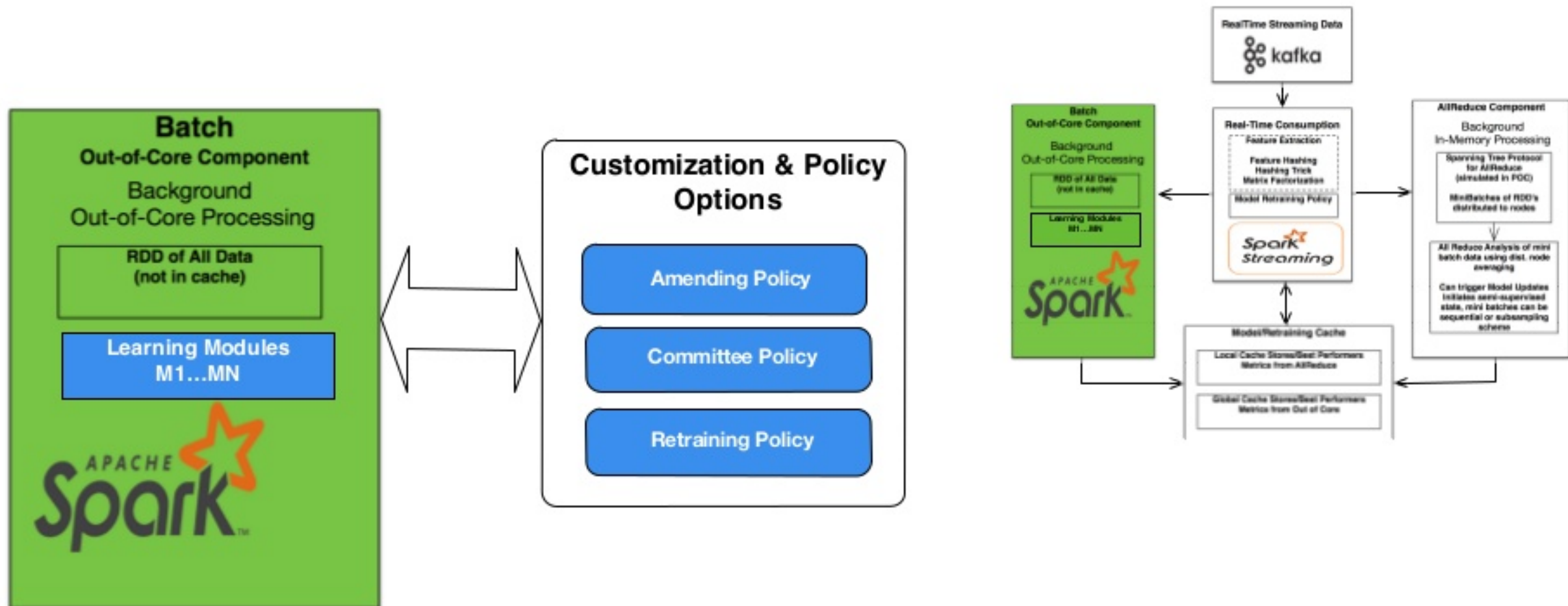
We want to infer a hypothesis using \mathcal{T} and use this hypothesis to predict labels we have not yet seen.

The semi-supervised learning case.

Online and Semi-supervised Learning: Semi-supervised Constraint



The Framework: Batch Component



The Framework: Batch Component

- Features
 - Out-of-Core, can run as a background process
 - Ensemble Learning: Multiple model/ Co-training learning algorithms
 - Amending Policy
 - Retrain Policy
 - Active Learning Policy
 - Custom Parameters

The Framework: Batch Component

CoTraining Algorithm #1

[Blum&Mitchell, 1998]

Given: labeled data L ,

unlabeled data U

Loop:

Train g_1 (hyperlink classifier) using L

Train g_2 (page classifier) using L

Allow g_1 to label p positive, n negative exams from U

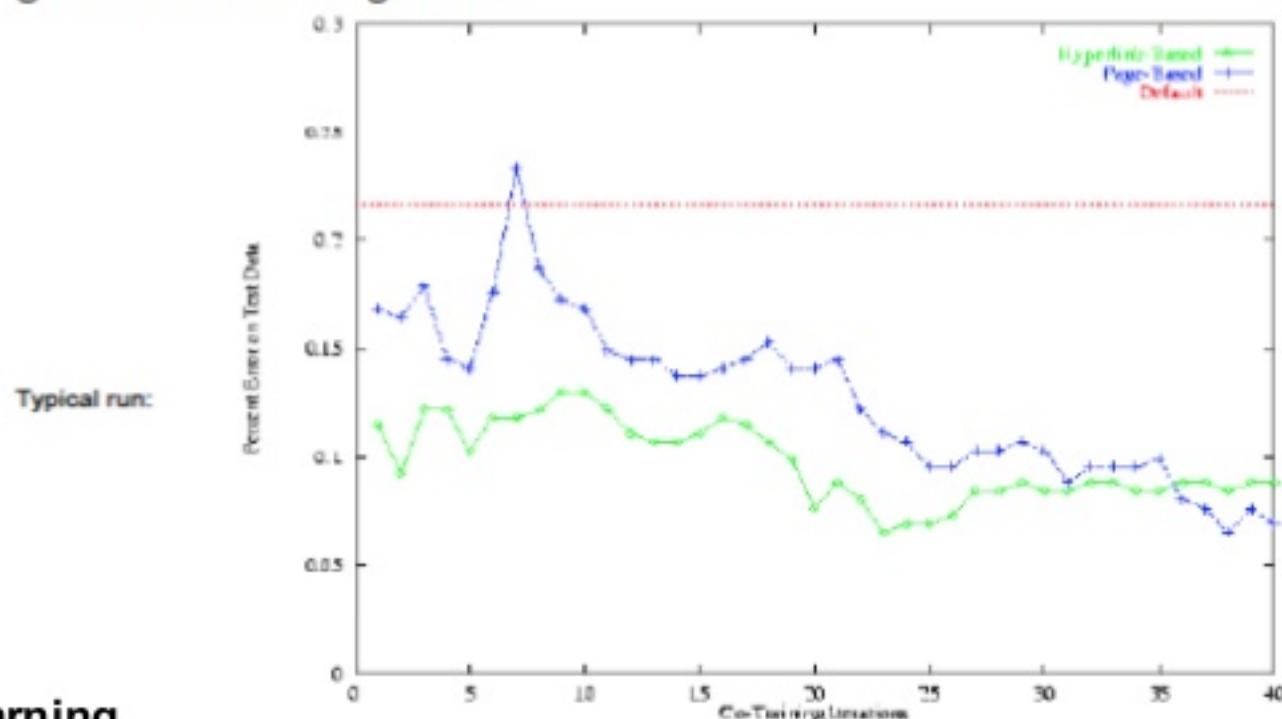
Allow g_2 to label p positive, n negative exams from U

Add these self-labeled examples to L

The Framework: Batch Component

CoTraining: Experimental Results

- begin with 12 labeled web pages (academic course)
- provide 1,000 additional unlabeled web pages
- average error: learning from labeled data 11.1%;
- average error: cotraining 5.0%



Semi-Supervised Learning

Edited by Olivier Chapelle, Bernhard Schölkopf and Alexander Zien

Learning from labeled and unlabeled data

Tom Mitchell, CMU

The Framework: Batch Component

- Multiple Model Learning/ Co-training
 - Expandable learning modules: multiple learning algorithms
 - Co-training for high-dimensionality, improved confidence
 - Accounts for different biases across learning techniques to strengthen the hypothesis
 - Improves confidence estimates in practice
 - Yields better results when there is a significant diversity among the models.
 - Reduces variance and helps to avoid overfitting

The Framework: Batch Component

Given a model, $m \in \mathcal{M}$

$$y_m(x) = h(x) + \epsilon_m(x) \quad (4)$$

We can get the average error over all models, $E_{average}$

$$E_{average} = \frac{1}{\mathcal{M}} \sum_{m=1}^{\mathcal{M}} \mathbb{E}_x[\epsilon_m(x)^2] \quad (5)$$

Regression: Modal averaging sum of squares/ Prediction confidence

Multiple Model-Based Reinforcement Learning

Kenji Doya, Kazuyuki Samejima, Ken-ichi Katagiri and Mitsuo Kawato



Vainsencher, Daniel, Shie Mannor, and Huan Xu. "Learning multiple models via regularized weighting." Advances in Neural Information Processing Systems. 2013.

The Framework: Batch Component

- Amending Policy
 - Instances are added sequentially and maintain order
 - Only the most 'confident' predictions are added
 - If validation is received, inaccurate predictions are removed and retrained (w/n a threshold)
 - Low confidence, unlabeled data are stored for future validation or rescoring

The Framework: Batch Component



Amending Policy



Training Batch

The Framework: Batch Component

- Retrain Policy
 - Loss Minimization
 - Increases in expected loss above threshold
 - Accuracy
 - Ratio/Number of incorrect predictions
 - Schedule
 - Regular retraining schedule
 - Optimizations
 - Limited to a window over the given data set
 - Train over a specific time period to optimize for catastrophic events and concept shifts
 - Subsampling
 - Very large datasets can set a subsampling methodology to improve batch processing speed

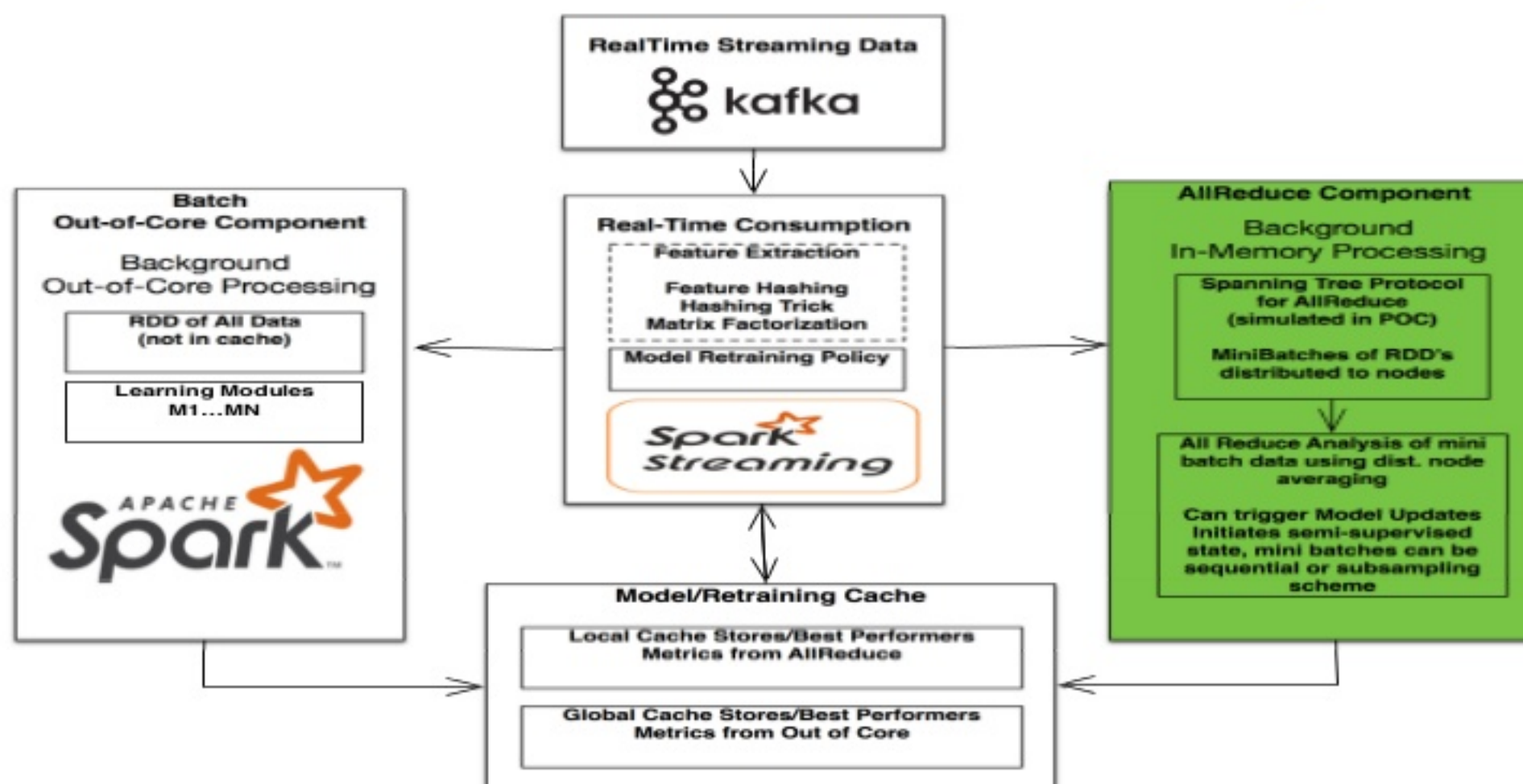
The Framework: Batch Component

- What happens to low confidence predictions?
- Active Learning Policy
 - Separate dataset holds these instances
 - Human labeling and/or an oracle is queried for accurate labels
- Google has over 10,000 contractors performing this function
 - <https://arstechnica.com/features/2017/04/the-secret-lives-of-google-raters/>
- Facebook is hiring 3000 new content monitors for a job AI cannot do
 - <http://www.popsoci.com/Facebook-hiring-3000-content-monitors>
- Netflix/Amazon are bringing in humans to improve the CTR recommendations
 - <http://blog.echen.me/2014/10/07/moving-beyond-ctr-better-recommendations-through-human-evaluation/>

The Framework: Batch Component

- Custom Parameters:
 - Window Size
 - Specify a time or range
 - Amending times
 - Schedule when amending policy runs
 - Default is automated at accuracy rates
 - Loss threshold
 - Specify maximum loss for a given loss function
 - Accuracy threshold
 - Specify the precision/recall rates before retraining is called on batch
 - Subsampling
 - Run a random subsample to train

The Framework: All-Reduce Component



The Framework: All-Reduce Component

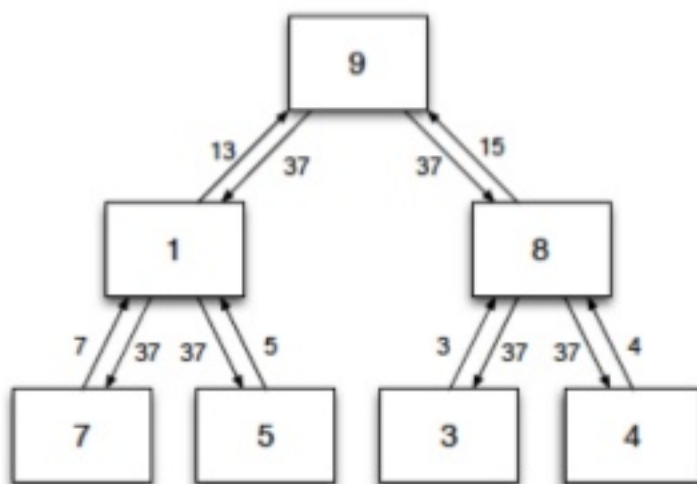
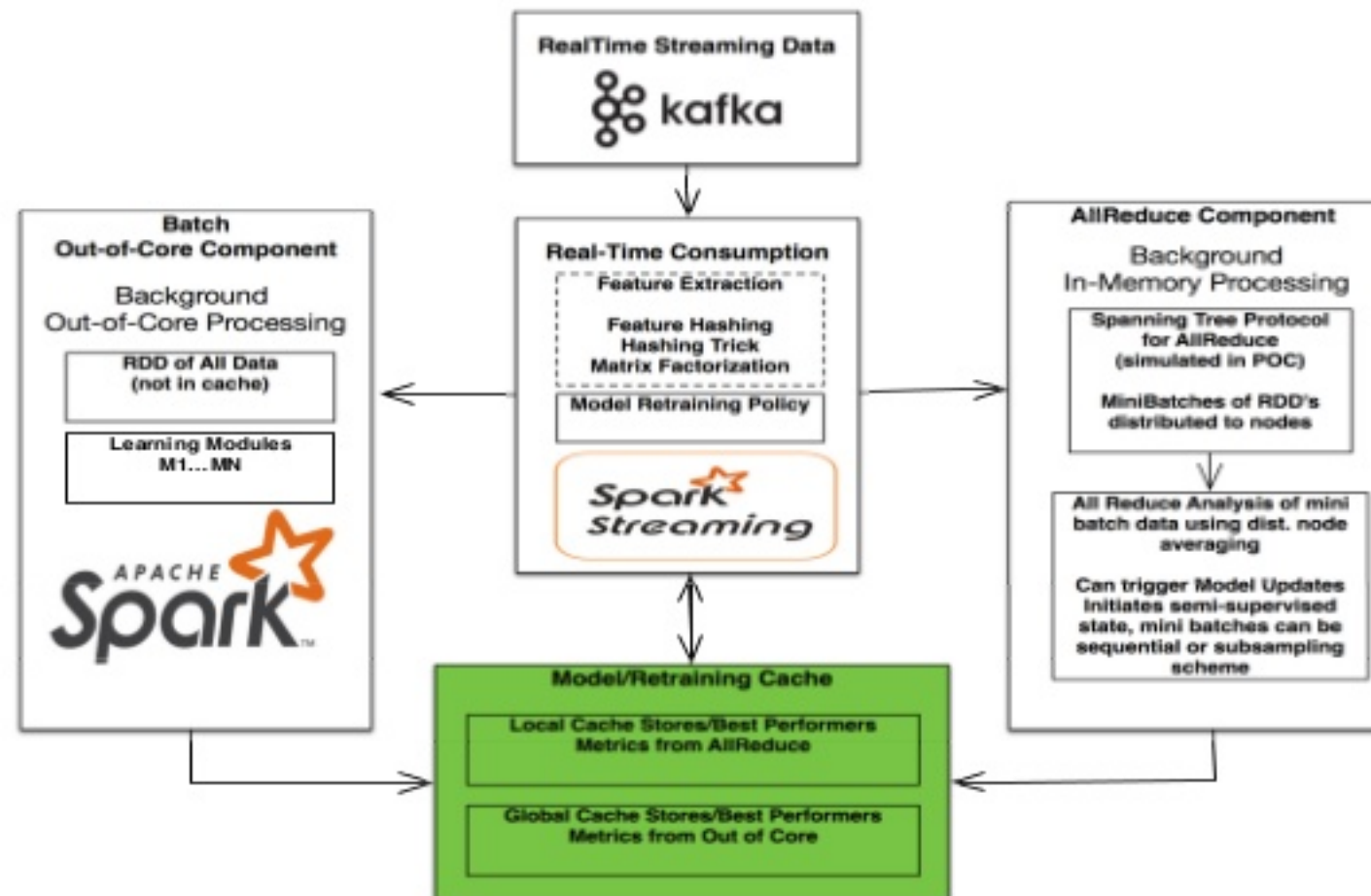


Figure 1: AllReduce operation. Initially, each node holds its own value. Values are passed up the tree and summed, until the global sum is obtained in the root node (reduce phase). The global sum is then passed back down to all other nodes (broadcast phase). At the end, each node contains the global sum.

- SGD/ LBFGS/Regression
- In-memory
- Minimal network latency
- Mini-batches of n/m for each nodes
- Head nodes holds an average of the weight parameters
- Can be added to existing implementations to optimize for online training

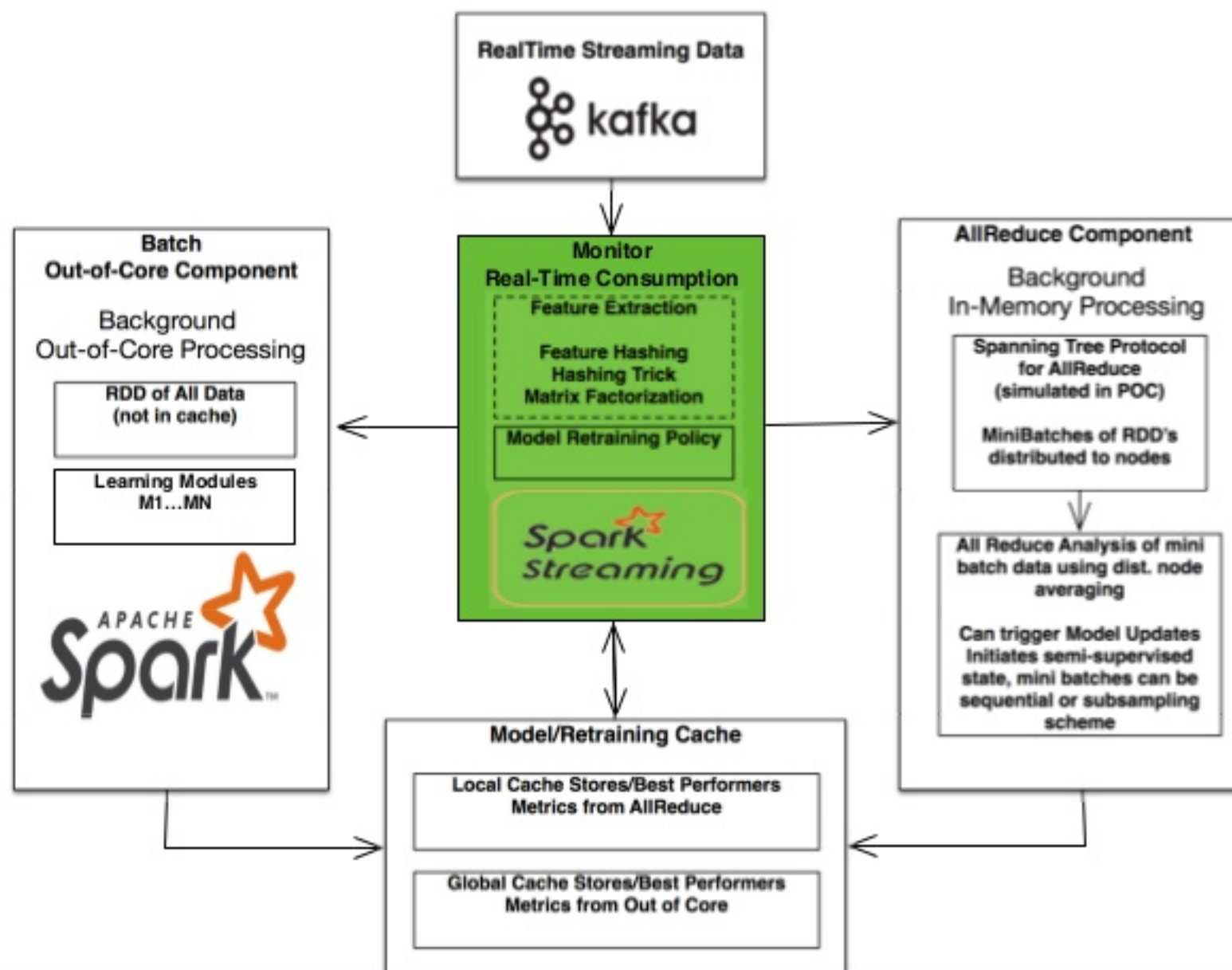
The Framework: Cache Component



The Framework: Cache Component

- Model Weights & Loss
- Custom Parameters
- When a the retraining policy is triggered in the monitor
 - A new best performing weight vector is selected from the cache
 - The running model weights are updated and used in real-time predictions

The Framework: Monitor Component



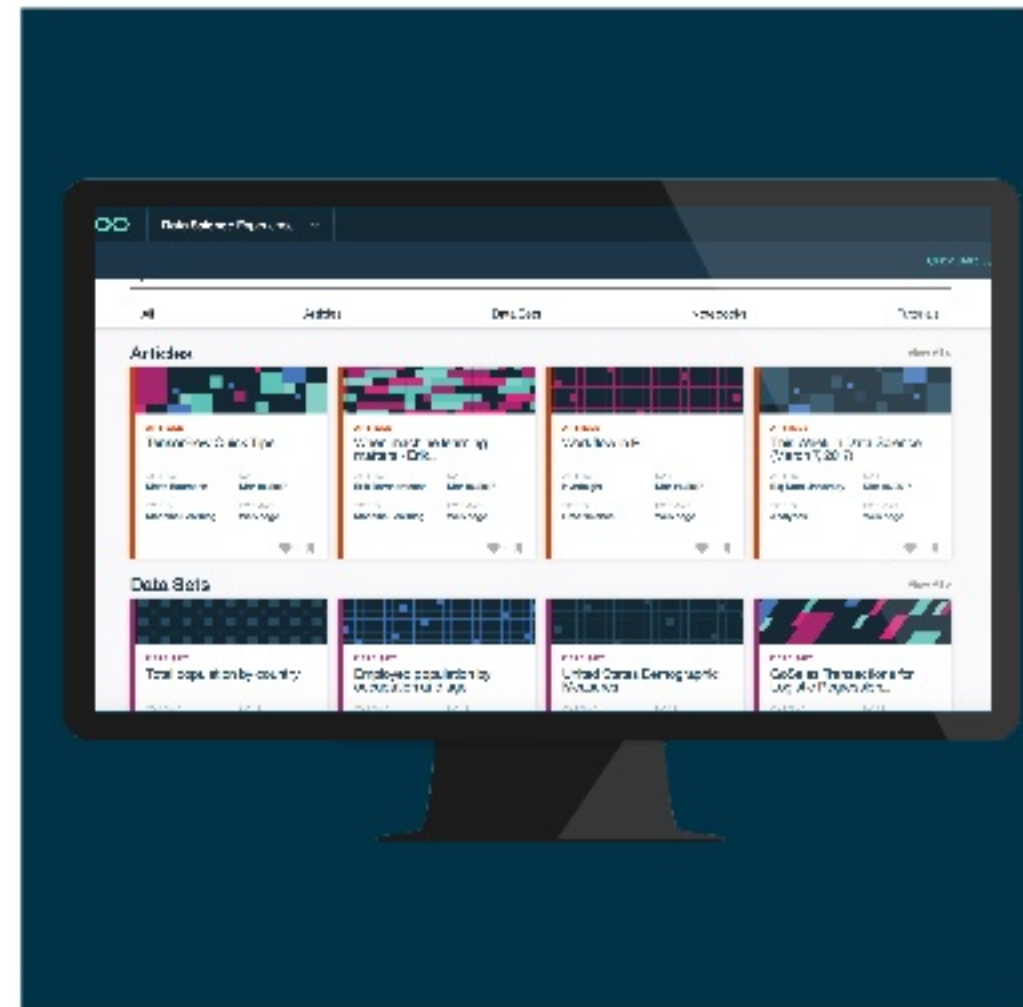
The Framework: Monitor Component

- Retraining in All-Reduce happens after each online pass
- Option 1: Makes real-time prediction with only local latency when initiated (Model Retrain Policy)
 - very low latency
 - stable predictions
 - Significant change in weights since last pass
 - Loss Minimization
 - Increases in expected loss above threshold
 - Accuracy
- Option 2: Can opt to use All-Reduce weights after the online pass
 - Low Latency
 - Greater sensitivity
 - Periodic queries to cache for best weights (Model Retrain Policy)

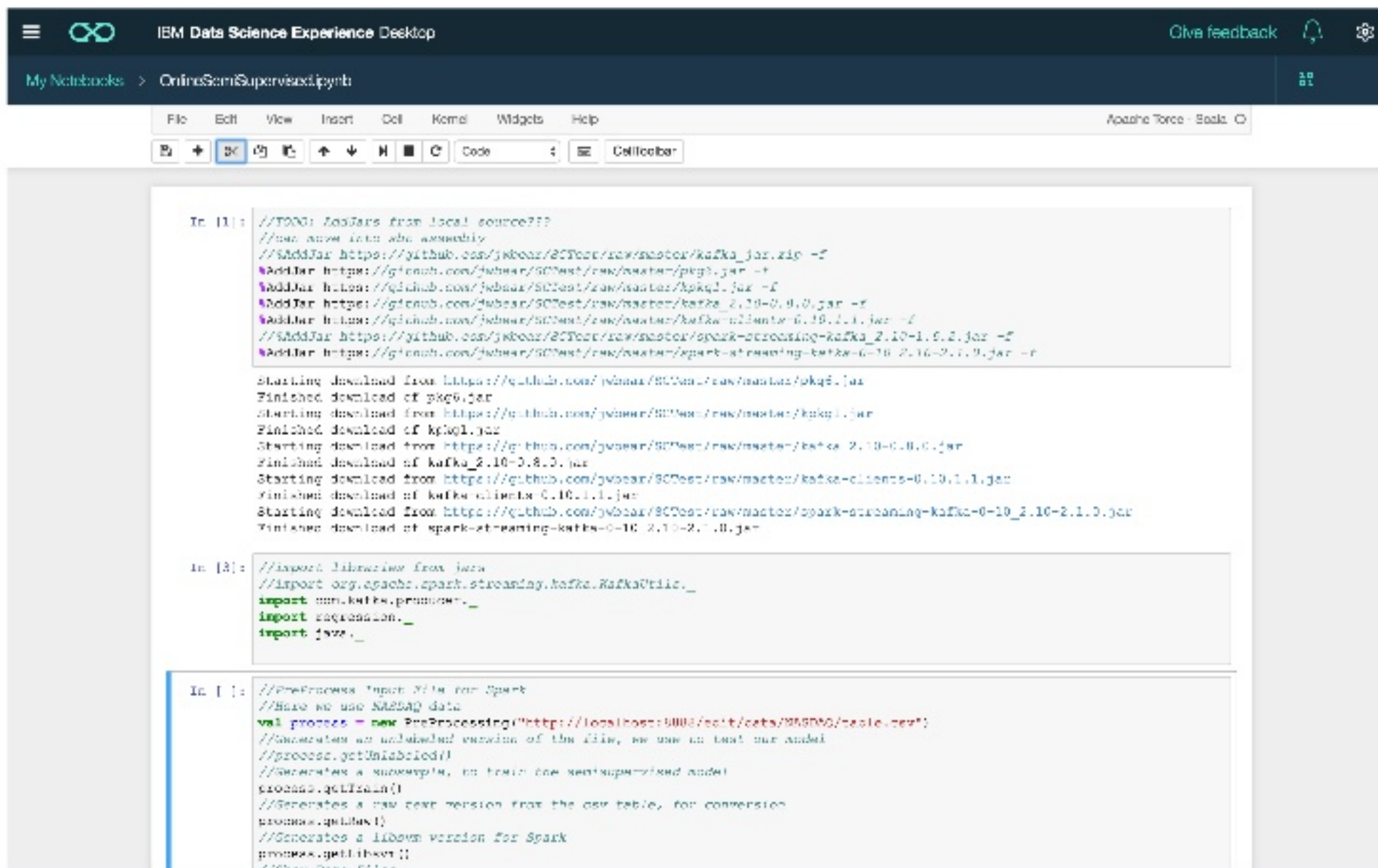
Performance: IBM's Data Science Experience

IBM Data Science Experience is an environment that brings together everything that a Data Scientist needs.

It includes the most popular **Open Source tools** such as Code in Scala/Python/R/SQL, Jupyter Notebooks, RStudio IDE and Shiny apps, Apache Spark and IBM unique value-add functionalities with **community** and **social features**, integrated as a first class citizen to make Data Scientists more successful.



Performance: IBM's Data Science Experience



The screenshot displays the IBM Data Science Experience Desktop interface. At the top, there is a navigation bar with the IBM logo, the text "IBM Data Science Experience Desktop", and links for "Give feedback", a notification bell, and a settings gear. Below this is a breadcrumb trail: "My Notebooks > OnlineSemiSupervised.ipynb". The main area shows a Jupyter Notebook with three code cells. The first cell contains a series of commands to download various JAR files from GitHub. The second cell imports necessary Python libraries. The third cell defines a preprocessing function for Spark. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations and execution.

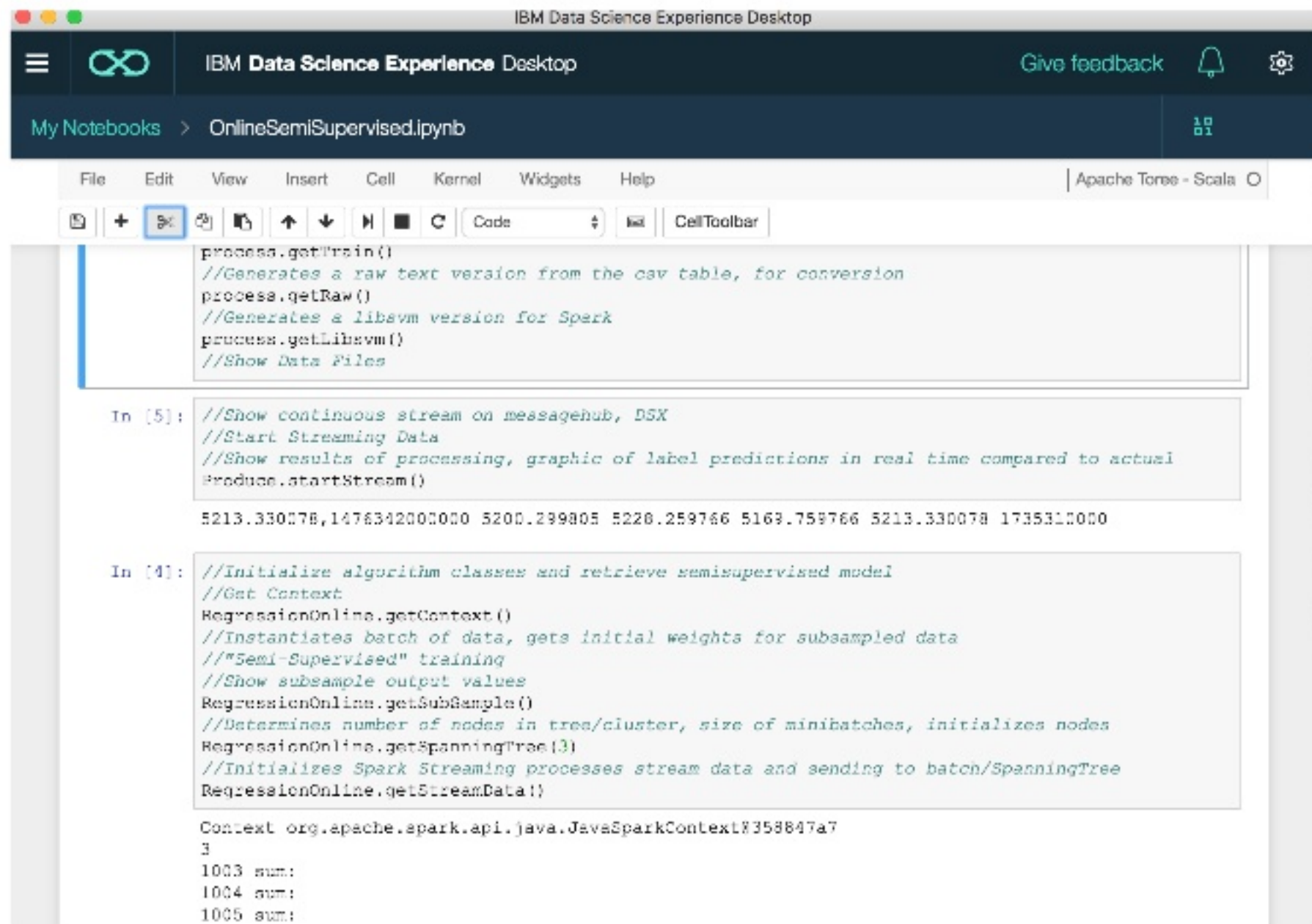
```
In [1]: //TODO: Load jars from local source???
//use above link aka assembly
//%AddJar https://github.com/jvbooz/SCTest/raw/master/kafka_jar.zip -f
%AddJar https://github.com/jvbooz/SCTest/raw/master/pkg9.jar -f
%AddJar https://github.com/jvbooz/SCTest/raw/master/pkg1.jar -f
%AddJar https://github.com/jvbooz/SCTest/raw/master/kafka_2.10-0.9.0.jar -f
%AddJar https://github.com/jvbooz/SCTest/raw/master/kafka-clients-0.10.1.1.jar -f
//%AddJar https://github.com/jvbooz/SCTest/raw/master/spark-streaming-kafka_2.10-1.0.2.jar -f
%AddJar https://github.com/jvbooz/SCTest/raw/master/spark-streaming-kafka-0-10_2.10-2.1.0.jar -f

Starting download from https://github.com/jvbooz/SCTest/raw/master/pkg9.jar
Finished download of pkg9.jar
Starting download from https://github.com/jvbooz/SCTest/raw/master/pkg1.jar
Finished download of pkg1.jar
Starting download from https://github.com/jvbooz/SCTest/raw/master/kafka_2.10-0.9.0.jar
Finished download of kafka_2.10-0.9.0.jar
Starting download from https://github.com/jvbooz/SCTest/raw/master/kafka-clients-0.10.1.1.jar
Finished download of kafka-clients-0.10.1.1.jar
Starting download from https://github.com/jvbooz/SCTest/raw/master/spark-streaming-kafka-0-10_2.10-2.1.0.jar
Finished download of spark-streaming-kafka-0-10_2.10-2.1.0.jar

In [3]: //import libraries from here
//import org.apache.spark.streaming.kafka.KafkaUtils._
import com.kaffee.promoter._
import regression._
import java._

In [ ]: //PreProcess input file for Spark
//Here we use NASDAQ data
val process = new PreProcessing("http://localhost:8080/cats/NASDAQ/test.csv")
//Generates an unlabeled version of the file, we use to test our model
process.getUnlabeled()
//Generates a supervised, to train the semi-supervised model
process.getTrain()
//Generates a raw text version from the csv table, for conversion
process.getRaw()
//Generates a libsvm version for Spark
process.getLibsvm()
//Show Data Files
```


Performance: IBM's Data Science Experience



The screenshot displays the IBM Data Science Experience Desktop interface. At the top, there's a header bar with the IBM logo, the text "IBM Data Science Experience Desktop", a "Give feedback" link, and a settings icon. Below this is a breadcrumb navigation bar showing "My Notebooks > OnlineSemiSupervised.ipynb". A menu bar includes "File", "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help". To the right of the menu bar is a "Cell Toolbar" with icons for running, saving, and other cell actions. The main area shows a Jupyter notebook with three code cells. The first cell contains code for generating raw text and libsvm versions of data. The second cell, labeled "In [5]:", shows code for starting a streaming data process, with output displaying a long numerical string. The third cell, labeled "In [4]:", shows code for initializing algorithm classes and retrieving a semisupervised model, with output showing the Spark context and some summary statistics.

```
process.getTrain()
//Generates a raw text version from the csv table, for conversion
process.getRaw()
//Generates a libsvm version for Spark
process.getLibsvm()
//Show Data Files

In [5]: //Show continuous stream on messagehub, DSX
//Start Streaming Data
//Show results of processing, graphic of label predictions in real time compared to actual
Produce.startStream()

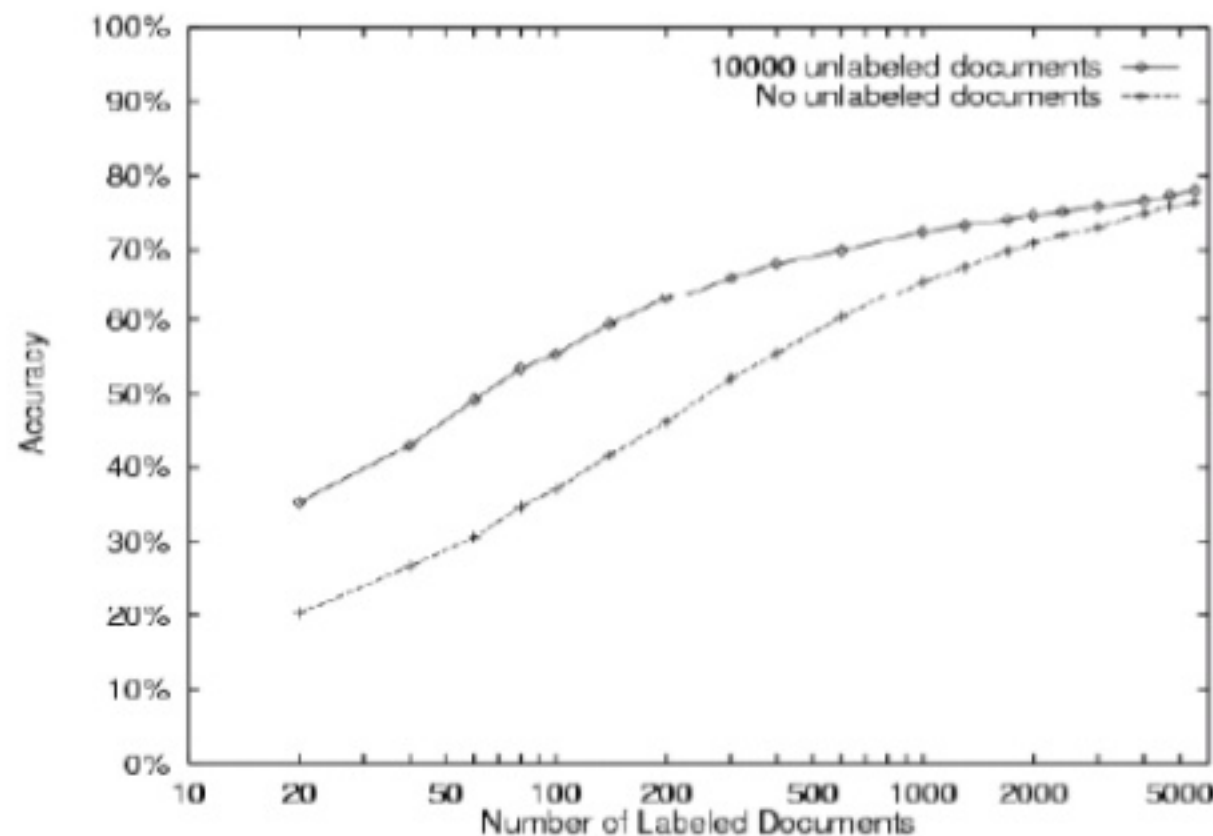
5213.330078,1476342000000 5200.299805 5228.259766 5169.759766 5213.330078 1735310000

In [4]: //Initialize algorithm classes and retrieve semisupervised model
//Get Context
RegressionOnline.getContext()
//Instantiates batch of data, gets initial weights for subsampled data
//"Semi-Supervised" training
//Show subsample output values
RegressionOnline.getSubSample()
//Determines number of nodes in tree/cluster, size of minibatches, initializes nodes
RegressionOnline.getSpanningTree(3)
//Initializes Spark Streaming processes stream data and sending to batch/SpanningTree
RegressionOnline.getStreamData()

Context org.apache.spark.api.java.JavaSparkContext17358847a7
3
1003 sum:
1004 sum:
1005 sum:
```


Performance: Accuracy Gains

20 Newsgroups



Important question! In many cases, unlabeled data is plentiful, labeled data expensive

- Medical outcomes ($x = \langle \text{symptoms}, \text{treatment} \rangle$, $y = \text{outcome}$)
- Text classification ($x = \text{document}$, $y = \text{relevance}$)
- Customer modeling ($x = \text{user actions}$, $y = \text{user intent}$)
- Sensor interpretation ($x = \langle \text{video}, \text{audio} \rangle$, $y = \text{who's there}$)

Semi-Supervised Learning

Edited by Olivier Chapelle, Bernhard Schölkopf and Alexander Zien

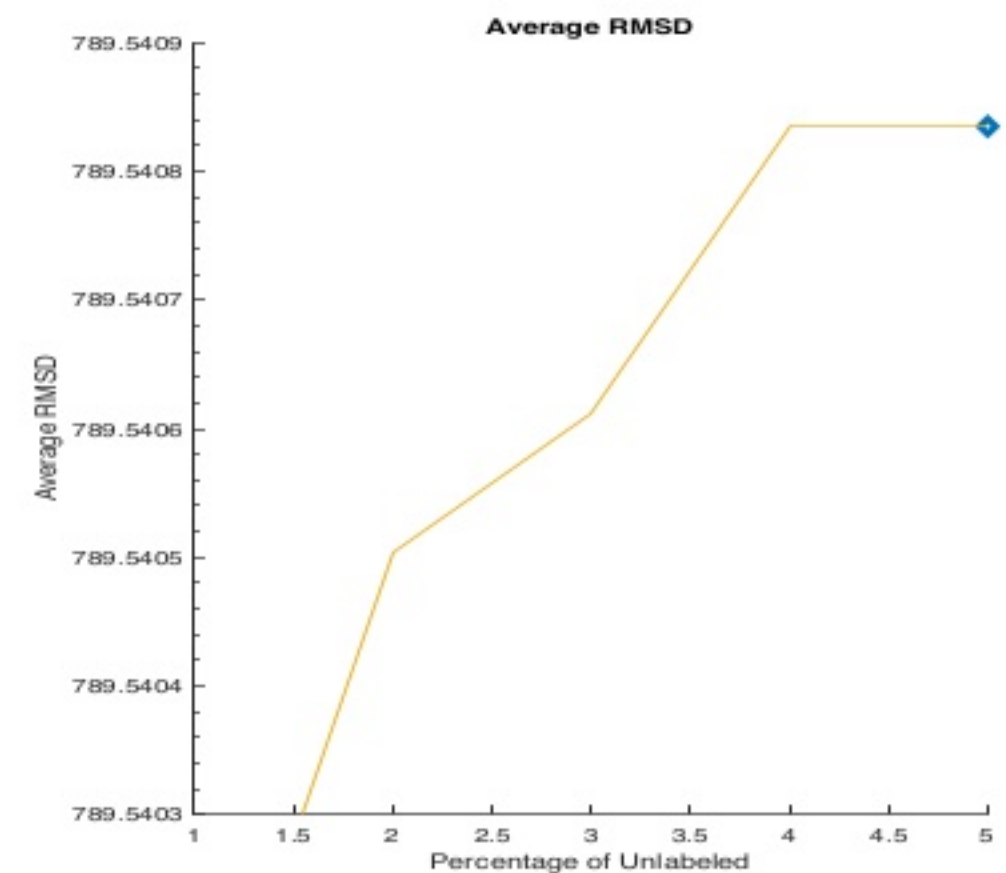
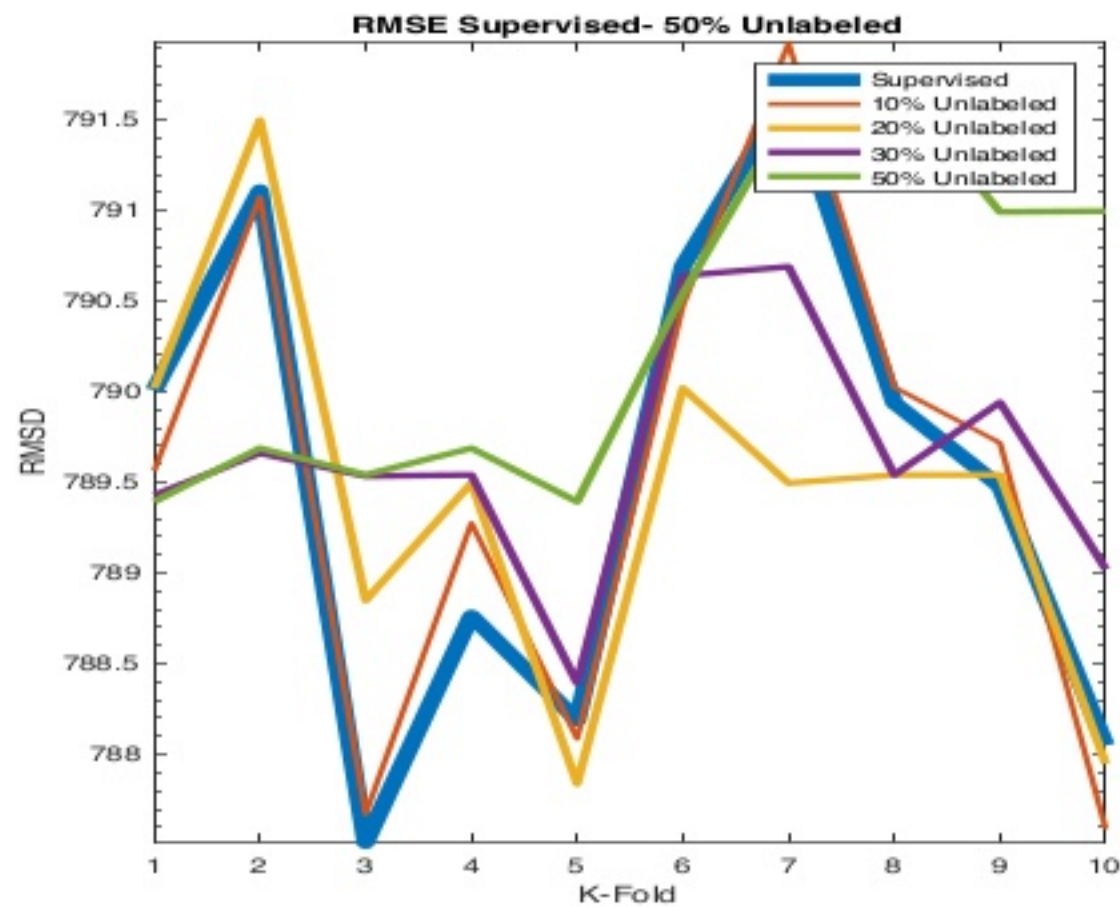
Learning from labeled and unlabeled data

Tom Mitchell, CMU

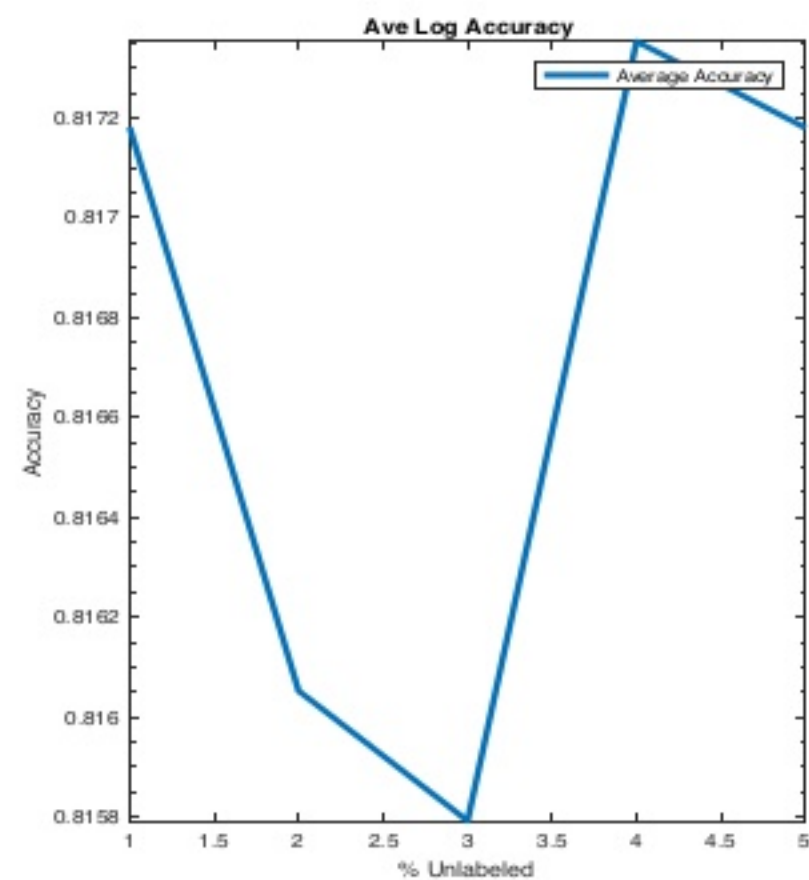
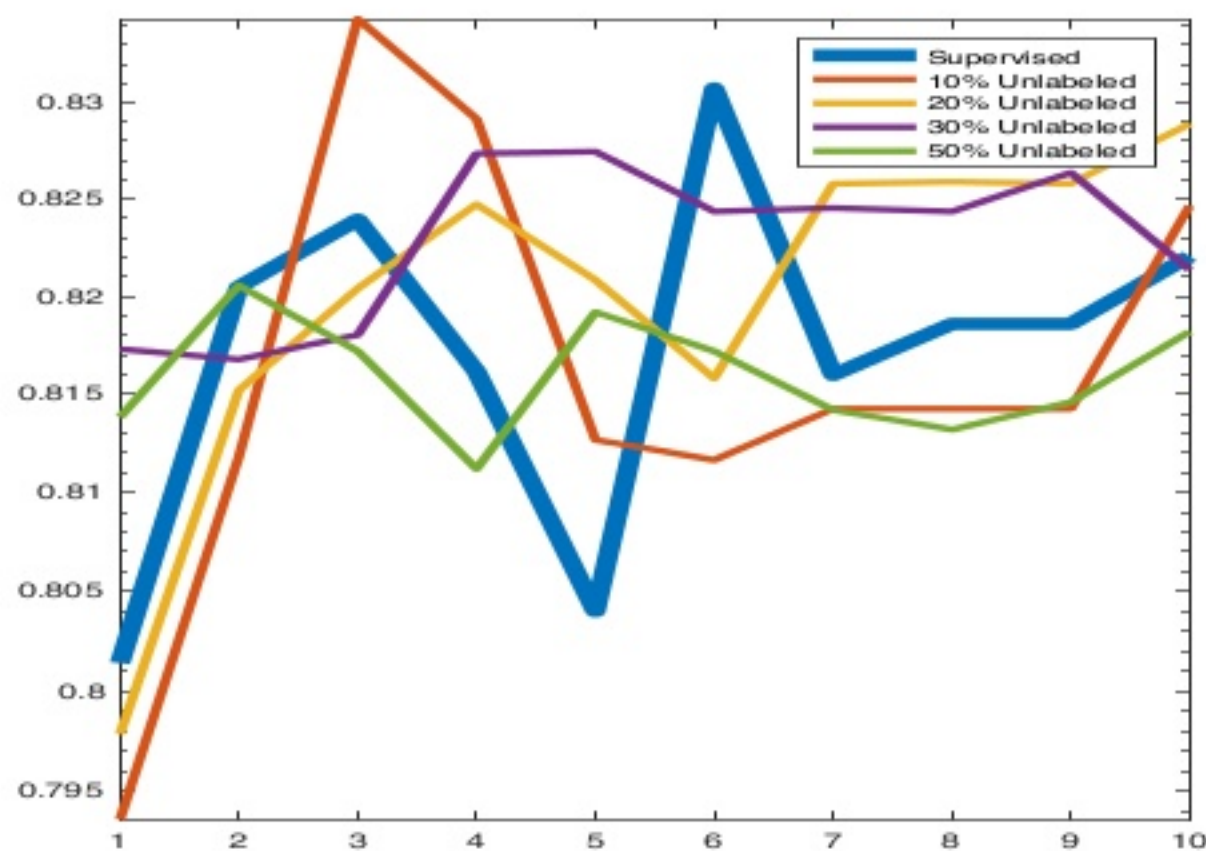
Performance: NASDAQ

- NASDAQ Data
 - Daily stock market values since 1971; ~50K instances
 - Features (excerpt)
 - Date, Open, High, Low, Close, Volume, Adj Close
 - Fully Labeled, $y = \text{close}$
 - k fold cross validation
 - Regression

Performance (1): Linear Regression



Performance (1A): Classification



Performance (2): Online & Semi-Supervised

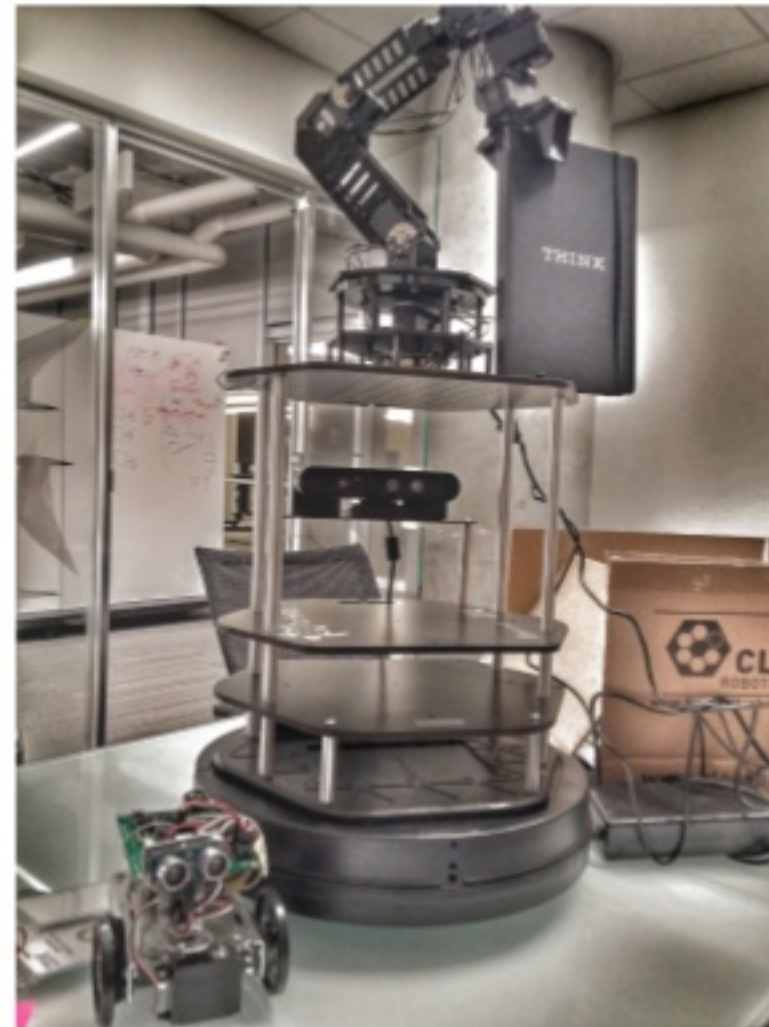
- HealthStats provides key health, nutrition and population statistics gathered from a variety of international sources
- Global Health Data since 1960; ~100K instances
 - Features (excerpt)
 - This dataset includes 345 indicators, such as immunization rates, malnutrition prevalence, and vitamin A supplementation rates across 263 countries around the world. Data was collected on a yearly basis from 1960-2016. Fully Labeled, $y = \text{close}$
 - k fold cross validation
 - Classification

Performance (3): Online Semi-Supervised Learning

- IBM Employee Attrition and Performance
 - Uncover the factors that lead to employee attrition
 - Features (excerpt)
 - Education 1 'Below College' 2 'College' 3 'Bachelor' 4 'Master' 5 'Doctor'
 - EnvironmentSatisfaction 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
 - JobInvolvement 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
 - JobSatisfaction 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
 - PerformanceRating 1 'Low' 2 'Good' 3 'Excellent' 4 'Outstanding'
 - RelationshipSatisfaction 1 'Low' 2 'Medium' 3 'High' 4 'Very High'
 - WorkLifeBalance 1 'Bad' 2 'Good' 3 'Better' 4 'Best'
 - Fully Labeled
 - k fold cross validation
 - Classification

Future Work

- More complex real-time data sets
- Real-Time Streaming and Semi-Supervised Learning for Autonomous Vehicles
- IoT and the Autonomous Vehicle in the Clouds: Simultaneous Localization and Mapping (SLAM) with Kafka and Spark Streaming (Spark Summit East 2017)
- Full Framework!!!



Future Work

- Improving Batch retraining policy to incorporate more information about the distributions of data
- Investigating model switching vs retraining
- Adding boosting mechanisms in batch
- Adding feature extraction for high dimensionality
- Expansion of Spark Streaming ML algorithms

Further Reading

A Reliable Effective Terascale Linear Learning System

Alekh Agarwal, Olivier Chapelle, Miroslav Dudik, John Langford

The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics) 2nd Edition by **Trevor Hastie (Author), Robert Tibshirani (Author), Jerome Friedman**

Semi-Supervised Learning

Edited by Olivier Chapelle, Bernhard Schölkopf and Alexander Zien

Learning from labeled and unlabeled data

Tom Mitchell, CMU

Multiple Model-Based Reinforcement Learning

Kenji Doya, Kazuyuki Samejima, Ken-ichi Katagiri and Mitsuo Kawato

Vainsencher, Daniel, Shie Mannor, and Huan Xu. "Learning multiple models via regularized weighting."

Advances in Neural Information Processing Systems. 2013.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory (COLT' 98)*. ACM, New York, NY, USA, 92-100.

DOI= <http://dx.doi.org/10.1145/279943.279962>



Thank You.

J. White Bear (jwhiteb@us.ibm.com)
IBM Spark Technology Center
505 Howard St San Francisco, CA