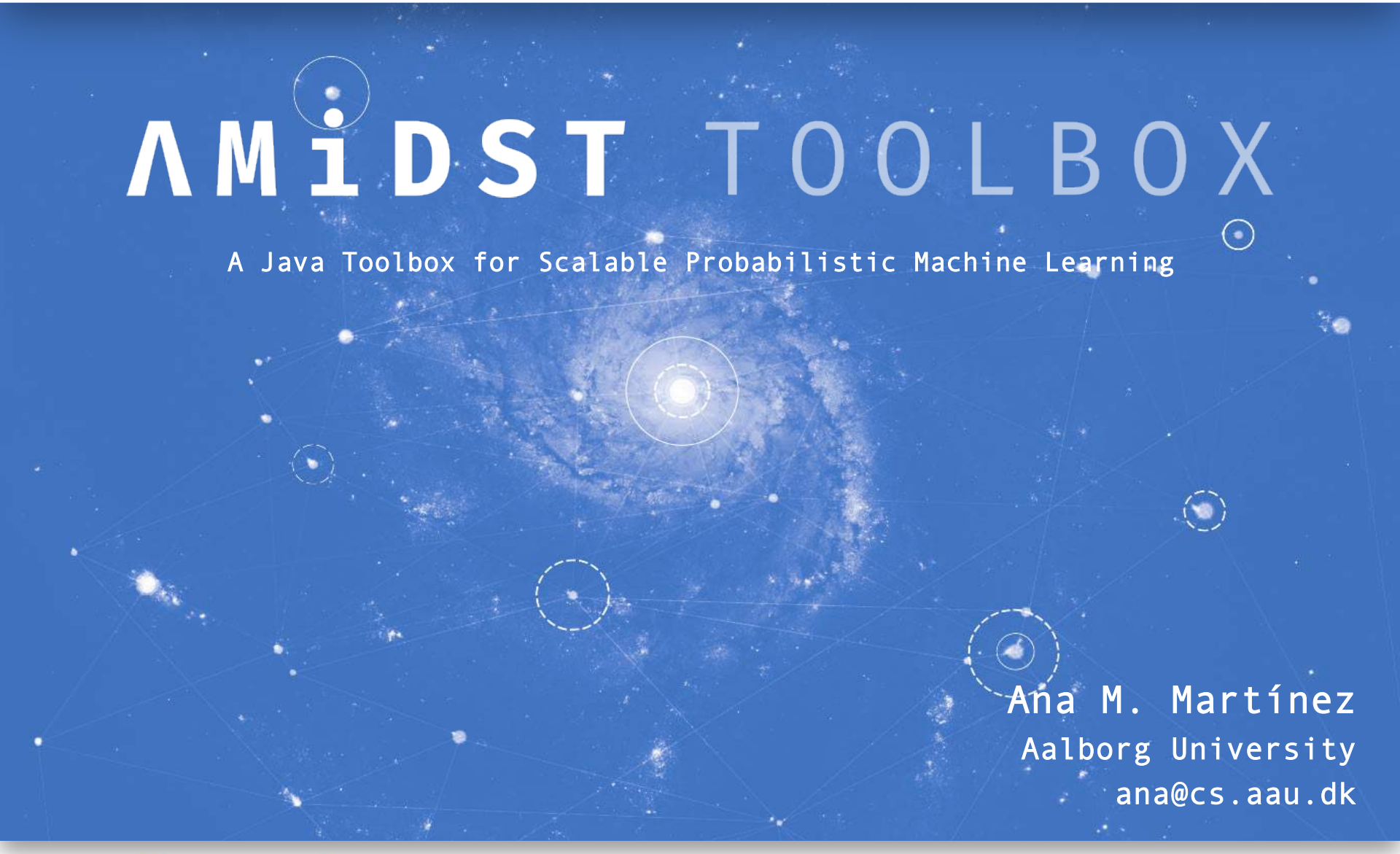


AMiDST TOOLBOX

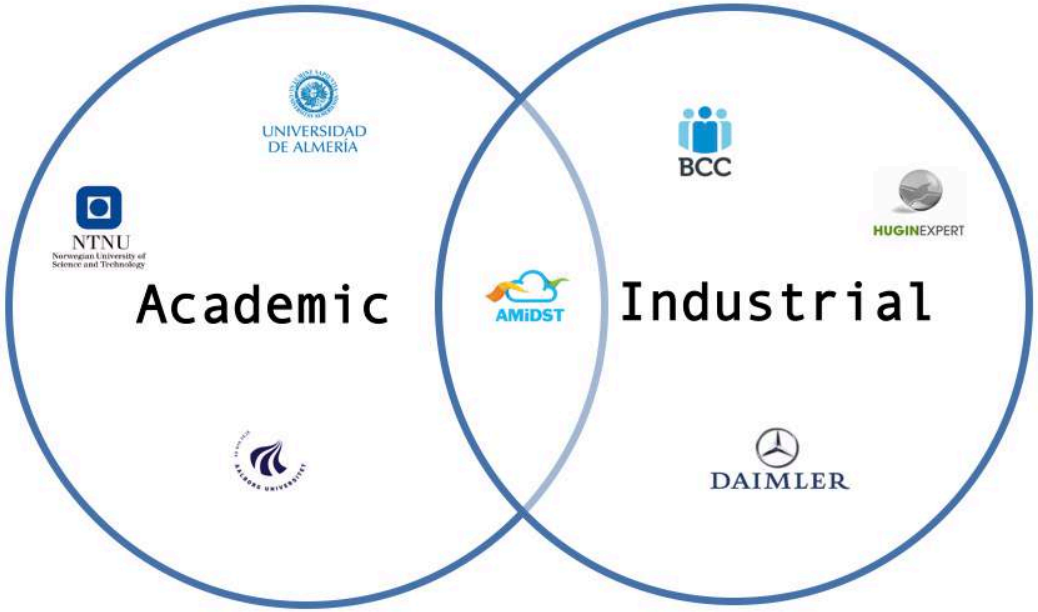
A Java Toolbox for Scalable Probabilistic Machine Learning

Ana M. Martínez
Aalborg University
ana@cs.aau.dk

Who are we?

THE AMIDST CONSORTIUM

AMiDST TOOLBOX



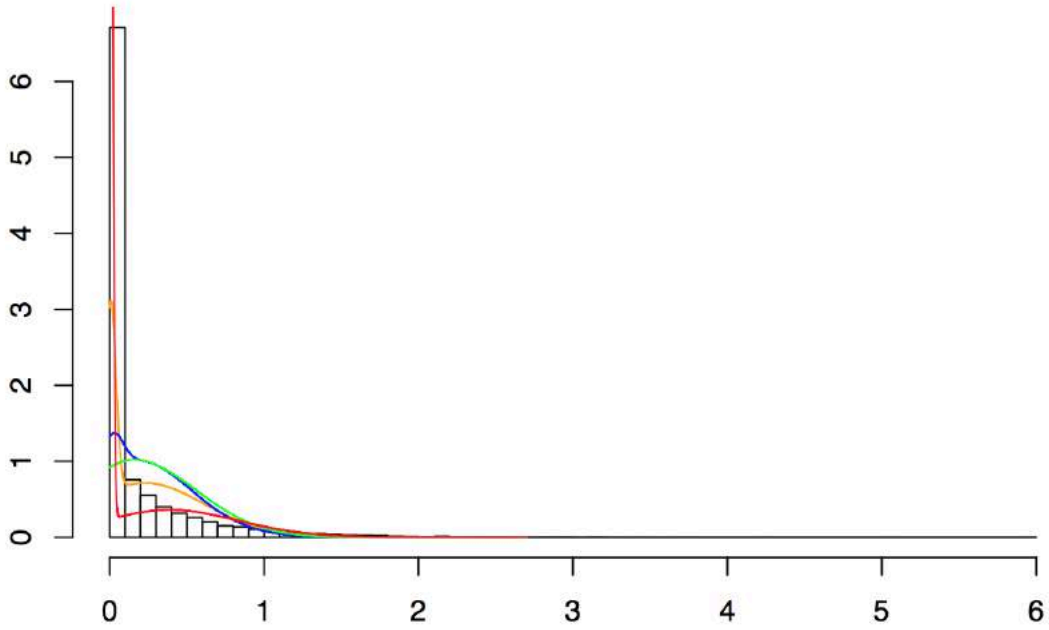
SEVENTH FRAMEWORK PROGRAMME

Running Use Case

Predicting Defaulting Clients

Predicts probability a customer will default within 2 years

Running Use Case



- Daily data for millions of clients
- Tons of missing data.
- Odd distributions.

Toolbox presentation

General Description

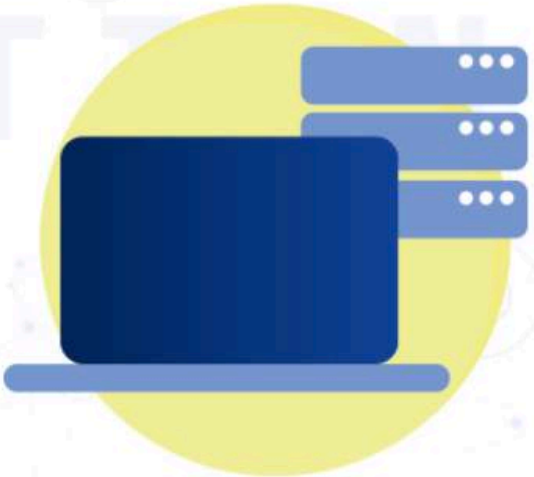
AMiDST TOOLBOX

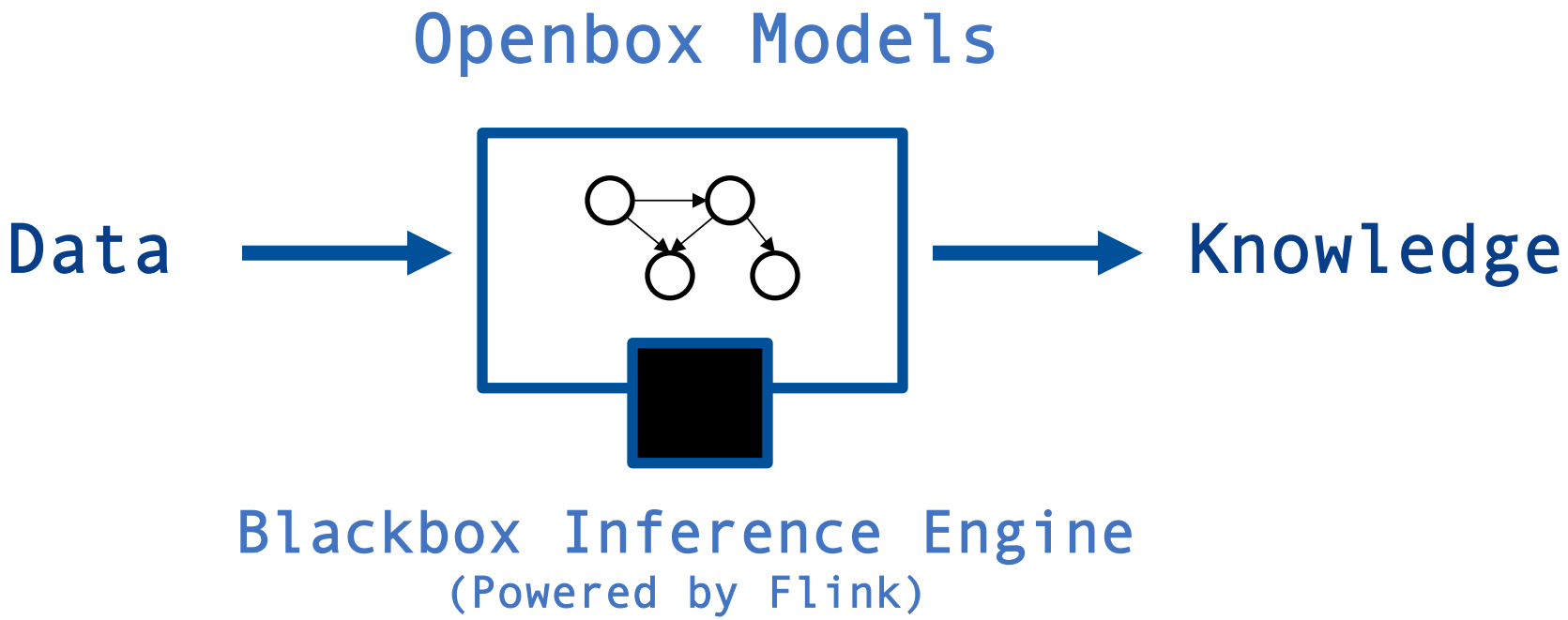
Probabilistic machine learning

Model your problem using a flexible probabilistic language based on graphical models. Then, fit it with data using a Bayesian approach to handle modelling uncertainty.

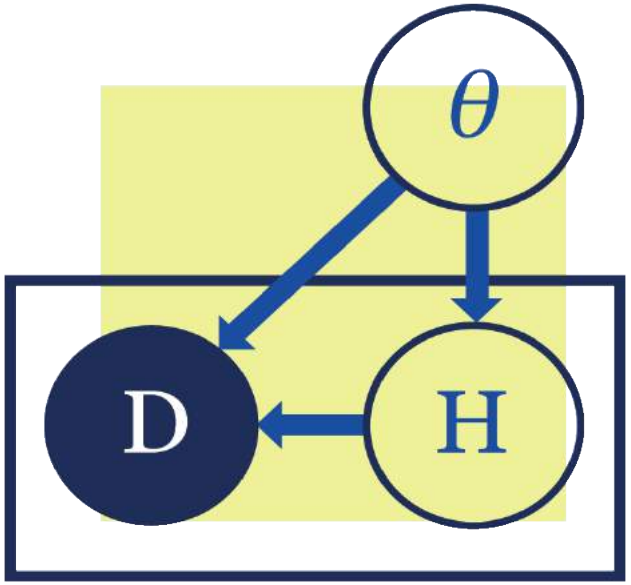
Multi-core and distributed processing

AMIDST provides tailored parallel and distributed implementations of Bayesian parameter learning (and probabilistic inference) for batch and streaming data. This processing is based on flexible and scalable message passing algorithms.




Openbox Models
Data
Knowledge
Blackbox Inference Engine
(Powered by Flink)

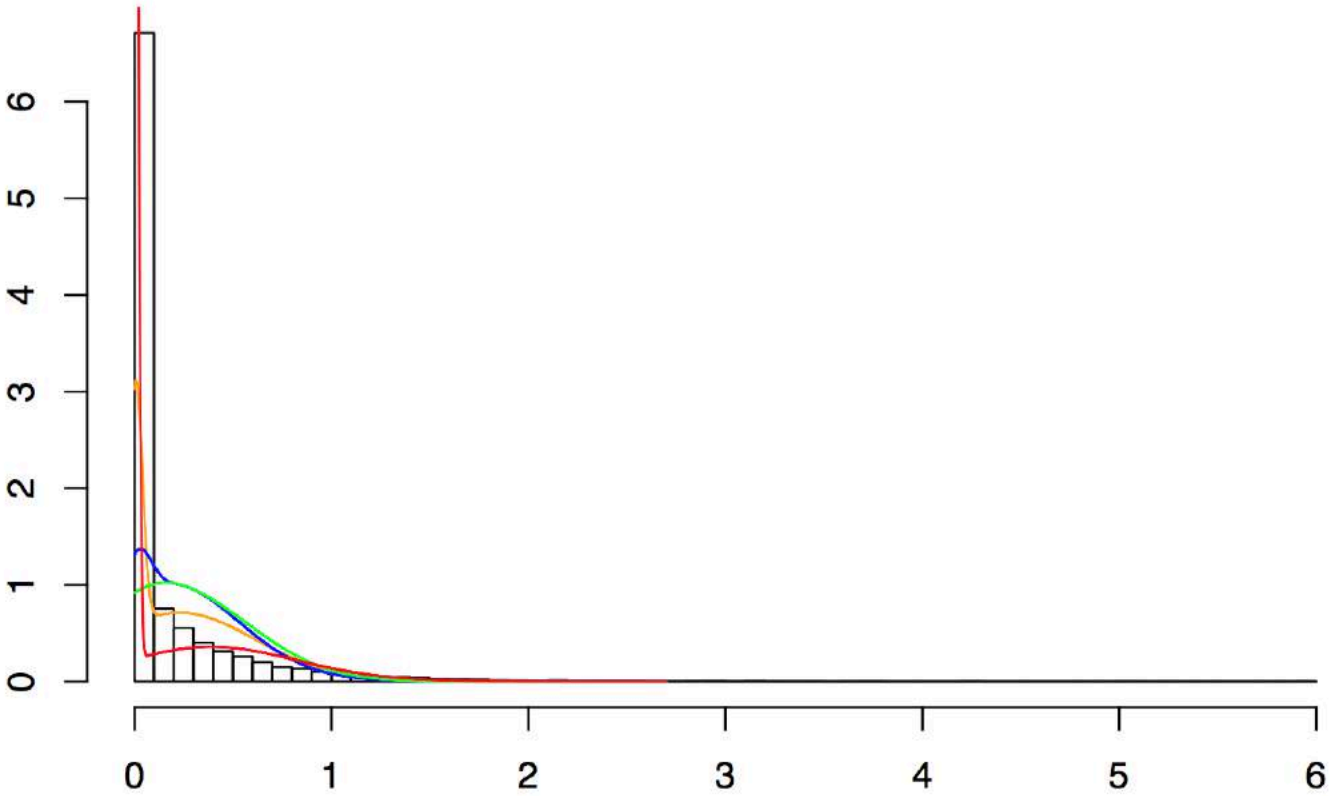
Main Features

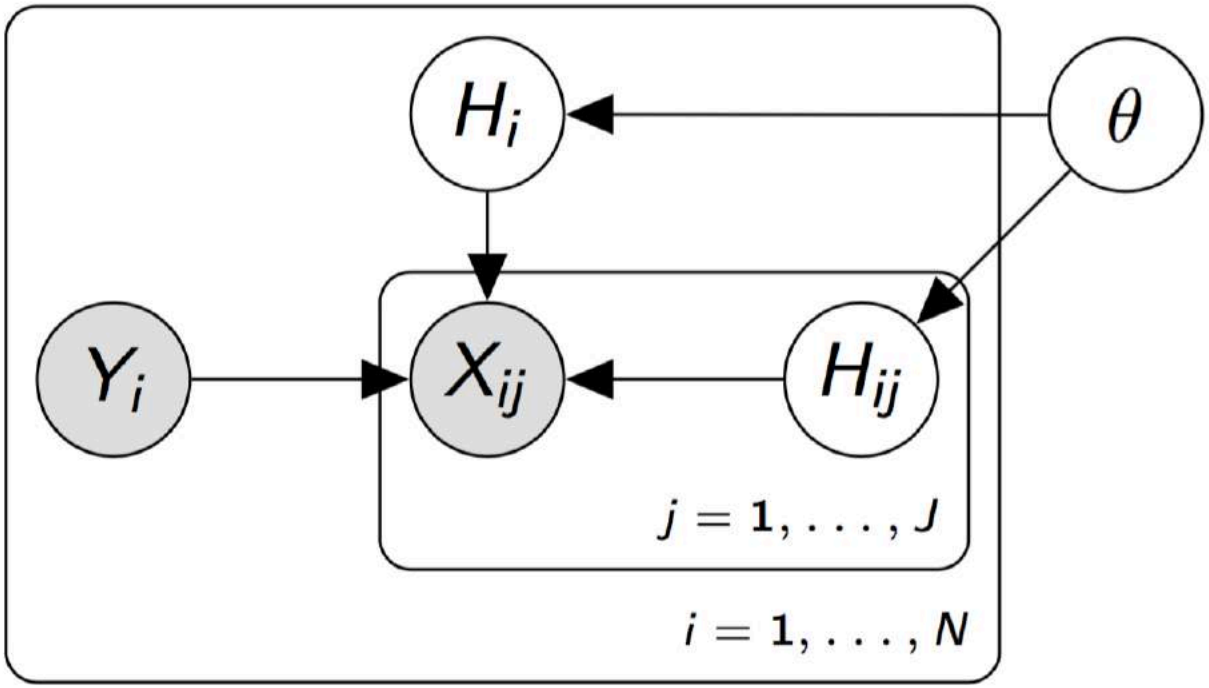


Probabilistic graphical models (PGMs)

Specify your model using probabilistic graphical models with latent variables and temporal dependencies

RUNNING USE CASE





Custom Gaussian Mixture Model

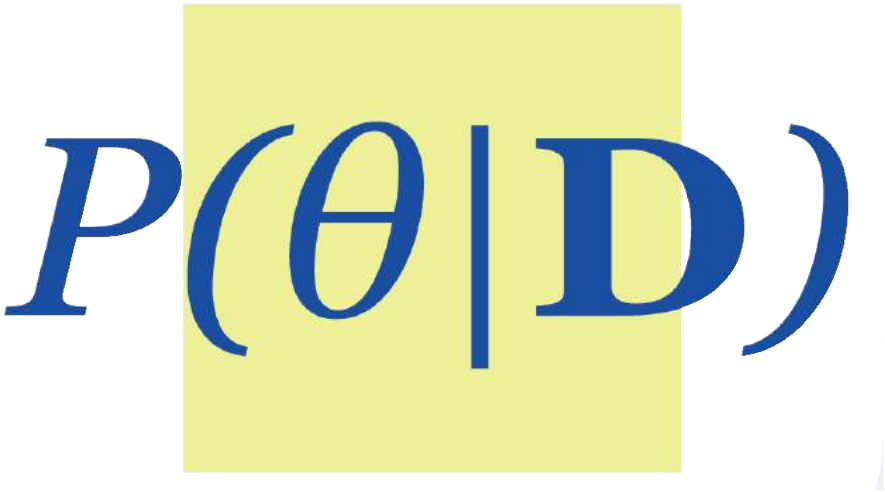
H_{ij} defines a local mixture.

H_i defines a global mixture.

```
//Set-up Flink session.
final ExecutionEnvironment env =
        ExecutionEnvironment.getExecutionEnvironment();

//Load the data stream
String filename = "hdfs://dataFlink_month0.arff";
DataFlink<DataInstance> data =
        DataFlinkLoader.loadDataFromFolder(env, filename,
        false);

//Build the model
Model model = new CustomGaussianMixture(data.getAttributes());
```

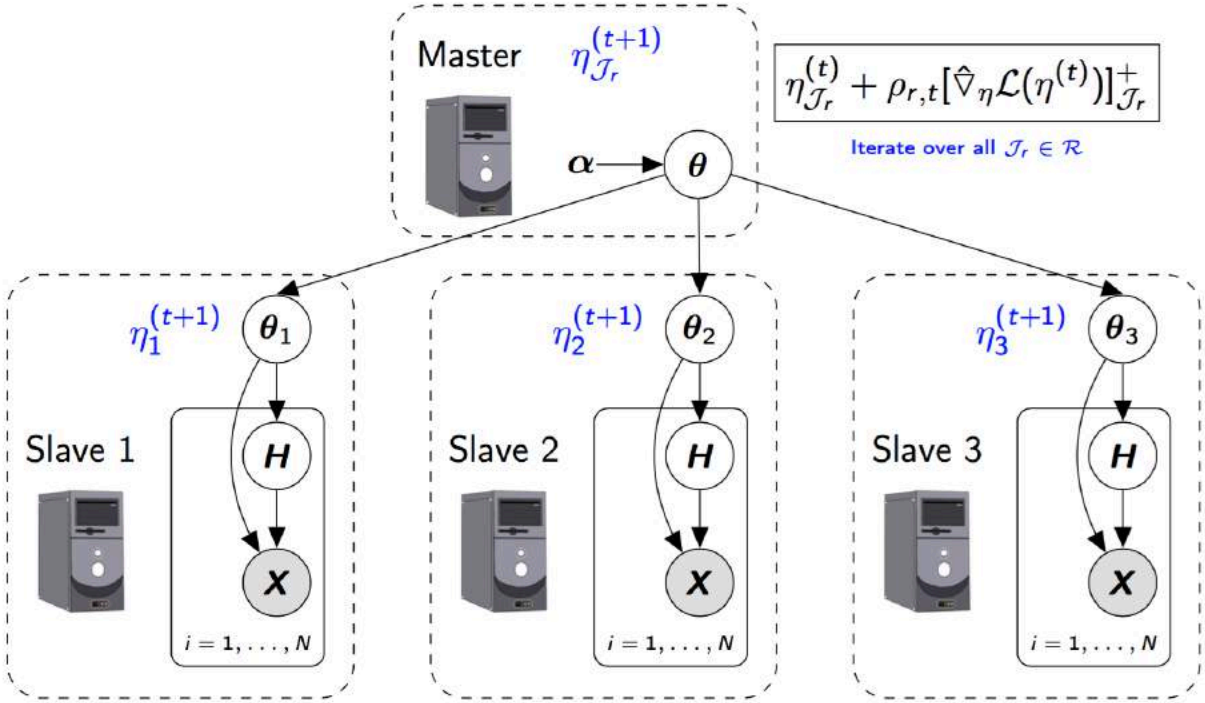


Scalable Learning

Perform Bayesian inference on your probabilistic models with powerful approximate and scalable algorithms.

SCALABLE INFERENCE

AMiDST TOOLBOX



d-VMP Algorithm - Coded as iterative map-reduce task

A state-of-the-art distributed variational message passing algorithm.

SCALABLE INFERENCE
RUNNING CODE EXAMPLE

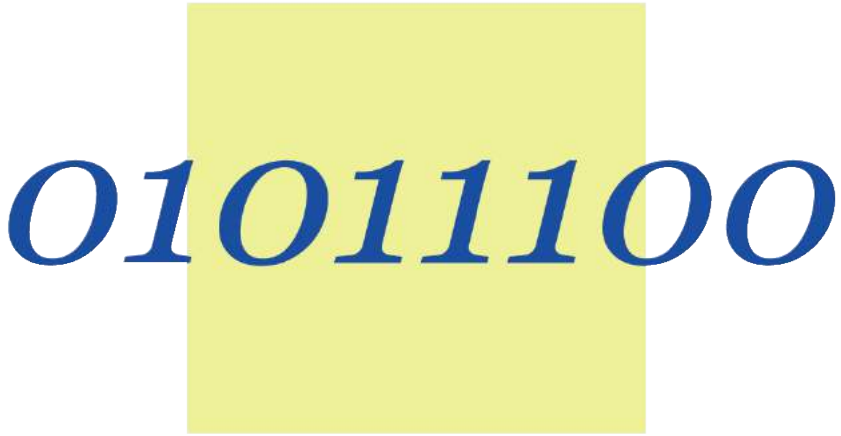
AMIDST TOOLBOX

```
//Set-up Flink session.
final ExecutionEnvironment env = ExecutionEnvironment.getExecutionEnvironment();

//Load the data stream
String filename = "hdfs://dataFlink_month0.arff";
DataFlink<DataInstance> data =
          DataFlinkLoader.loadDataFromFolder(env, filename, false);

//Build the model
Model model = new CustomGaussianMixture(data.getAttributes());


//Learn the model
model.updateModel(data);
```



Data Streams

Update your models when new data is available. This makes our toolbox appropriate for learning from data streams.

```
//Set-up Flink session.
final ExecutionEnvironment env = ExecutionEnvironment.getExecutionEnvironment();

//Load the data stream
String filename = "hdfs://dataFlink_month0.arff";
DataFlink<DataInstance> data =
          DataFlinkLoader.loadDataFromFolder(env, filename, false);

//Build the model
Model model = new CustomGaussianMixture(data.getAttributes());

//Learn the model
model.updateModel(data);

//Update your model
for(int i=1; i<12; i++) {
   filename = "dataFlink_month"+i+".arff";
   data = DataFlinkLoader.loadDataFromFolder(env, filename,false);
   model.updateModel(data);

}
```

Predicting Defaulting Clients

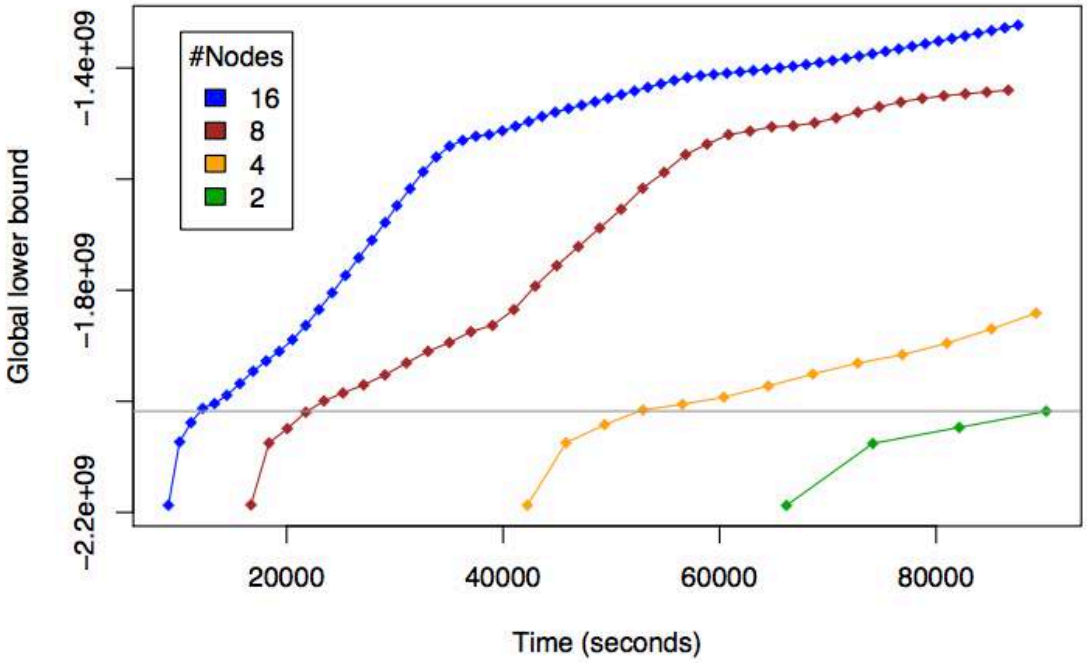
- Old BCC's models based on logistic regression got an AUC of 0.816.
- AMIDST's models gets an AUC of 0.952.

Scalability analysis

Use your defined models to process massive data sets in a distributed computer cluster using Flink.

SCALABILITY ANALYSIS

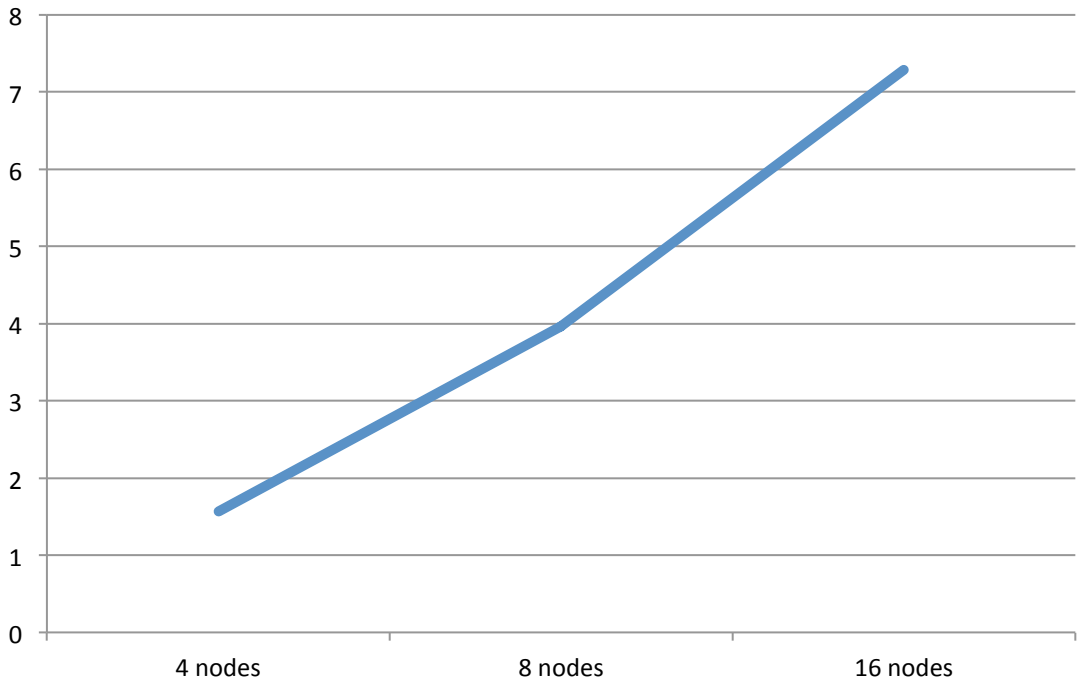
AMiDST TOOLBOX



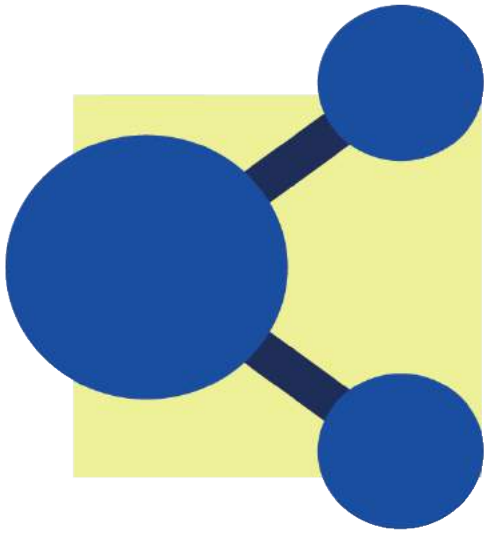
One billion node probabilistic model

Experiment on a Flink cluster with 16 nodes on AWS.

SCALABILITY ANALYSIS



- Speedup (with respect to 2 nodes)

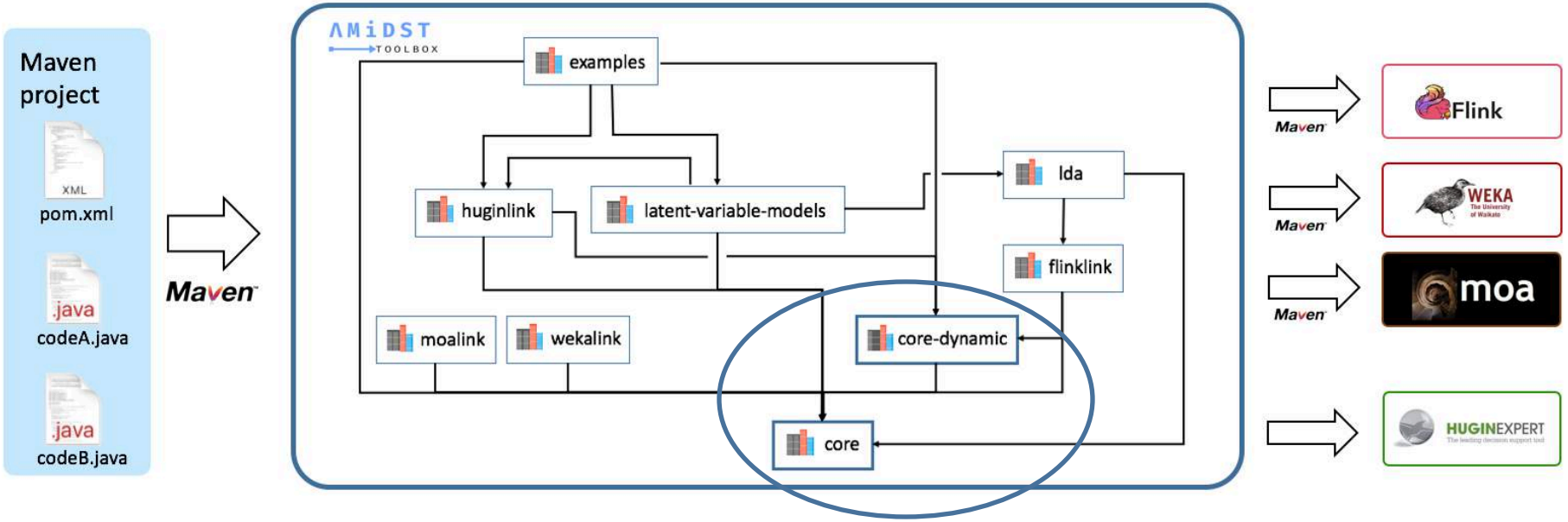


Modular Design

The AMIDST Toolbox has been designed following a modular structure. This makes easier:

- The maintenance and enhancement of the software
- The integration with external software: HUGIN, MOA, Weka, R.

MODULAR DESIGN



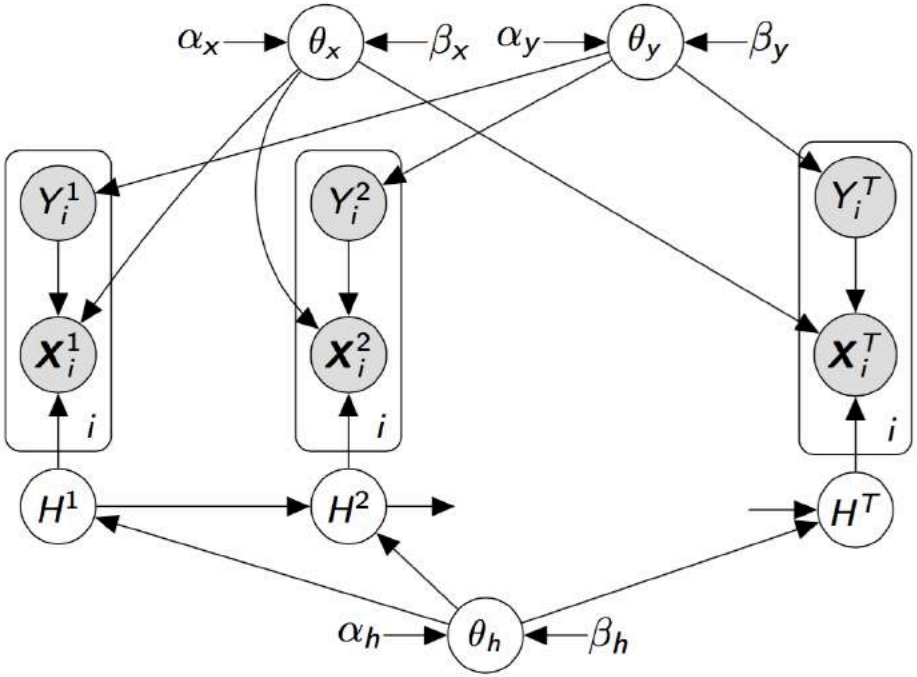
Running Use Case II

Tracking Concept Drift

Detects changes in customer profiles during Spanish financial crisis

Concept Drift Detection

Model



Hidden Variables are used to capture changes in customer profile

CONCEPT DRIFT DETECTION
RUNNING CODE

AMIDST TOOLBOX

```
//Set-up Flink session.
final ExecutionEnvironment env = ExecutionEnvironment.getExecutionEnvironment();

//Load the data stream
String filename = "hdfs://dataFlink_month0.arff";
DataFlink<DataInstance> data =
          DataFlinkLoader.loadDataFromFolder(env, filename, false);

//Build the model
Model model = new ConceptDriftDetector(data.getAttributes());

//Learn the model
model.updateModel(data);

//Update your model
for(int i=1; i<12; i++) {
   filename = "dataFlink_month"+i+".arff";
   data = DataFlinkLoader.loadDataFromFolder(env, filename,false);
   model.updateModel(data);
   System.out.println(model.getPosteriorDistribution("hiddenVar").
                                                        toString());
}
```

Concept Drift Detection

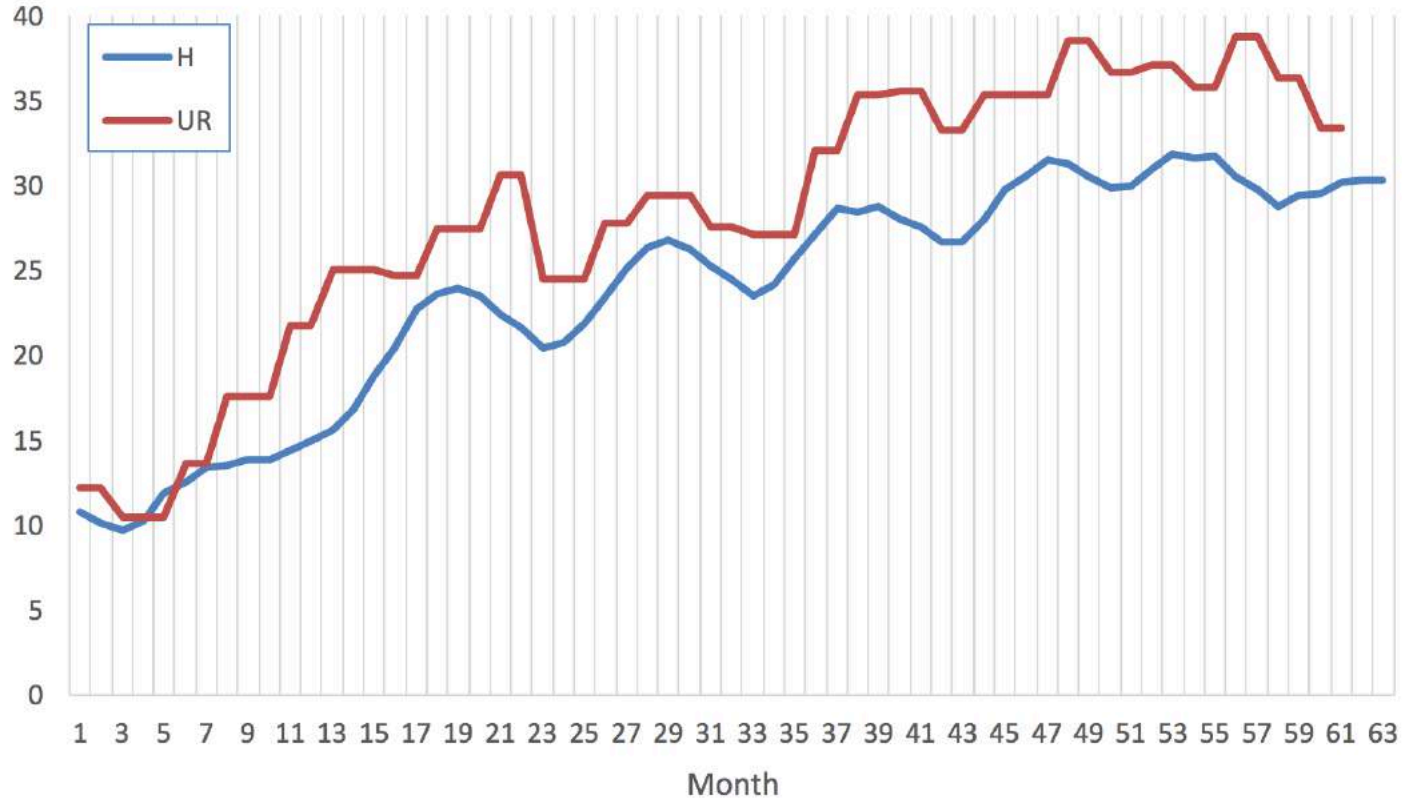
Results

Hidden Variable Captures Concept Drift

Drift Pattern: Seasonal + Global trend

Concept Drift Detection

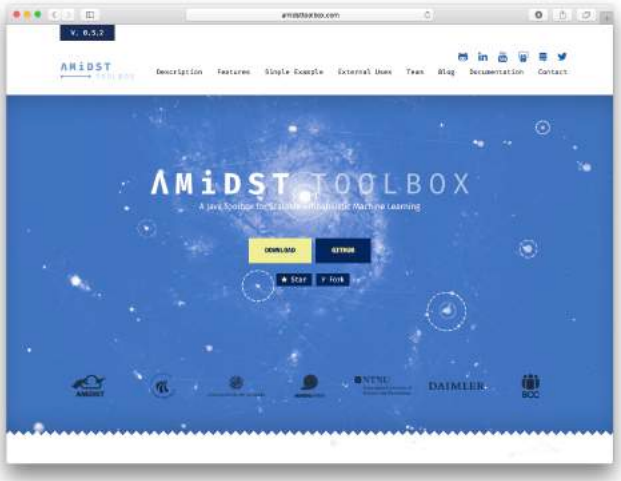
Results



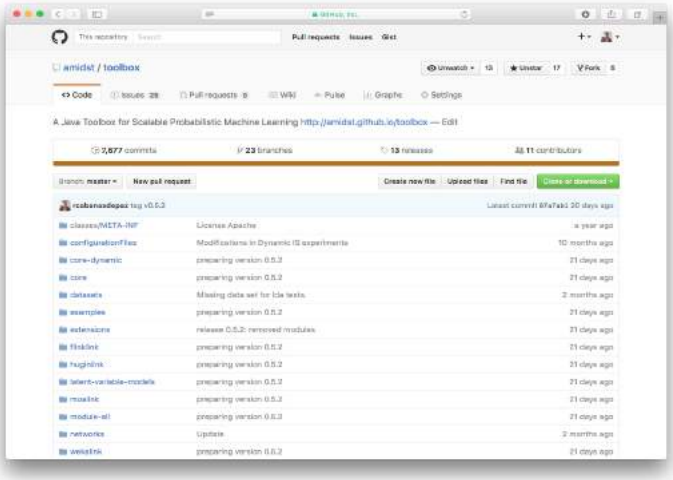
Unemployment Rate main driver of Concept Drift

Hidden Variable correlates with unemployment rate (rho = 0.961)

COLLABORATE



www.amidsttoolbox.com



github.com/amidst/toolbox

Appache
License 2.0

Thanks for your attention

www.amidsttoolbox.com

contact@amidsttoolbox.com

@AmidstToolbox