Continuous Applications using Apache Spark

Online Machine Learning with Streaming Data

Robert Hamlet
MSc Big Data

Supervised by Andreas Lang and Jonathan Forbes (Aquila Insight),
Dr. Kevin Swingler (University of Stirling)

Background

Streaming Data

Applying models to streaming data reduces 'time-to-insight', critical for time-critical applications of machine learning such as anomaly detection and e-commerce.

Online Machine Learning

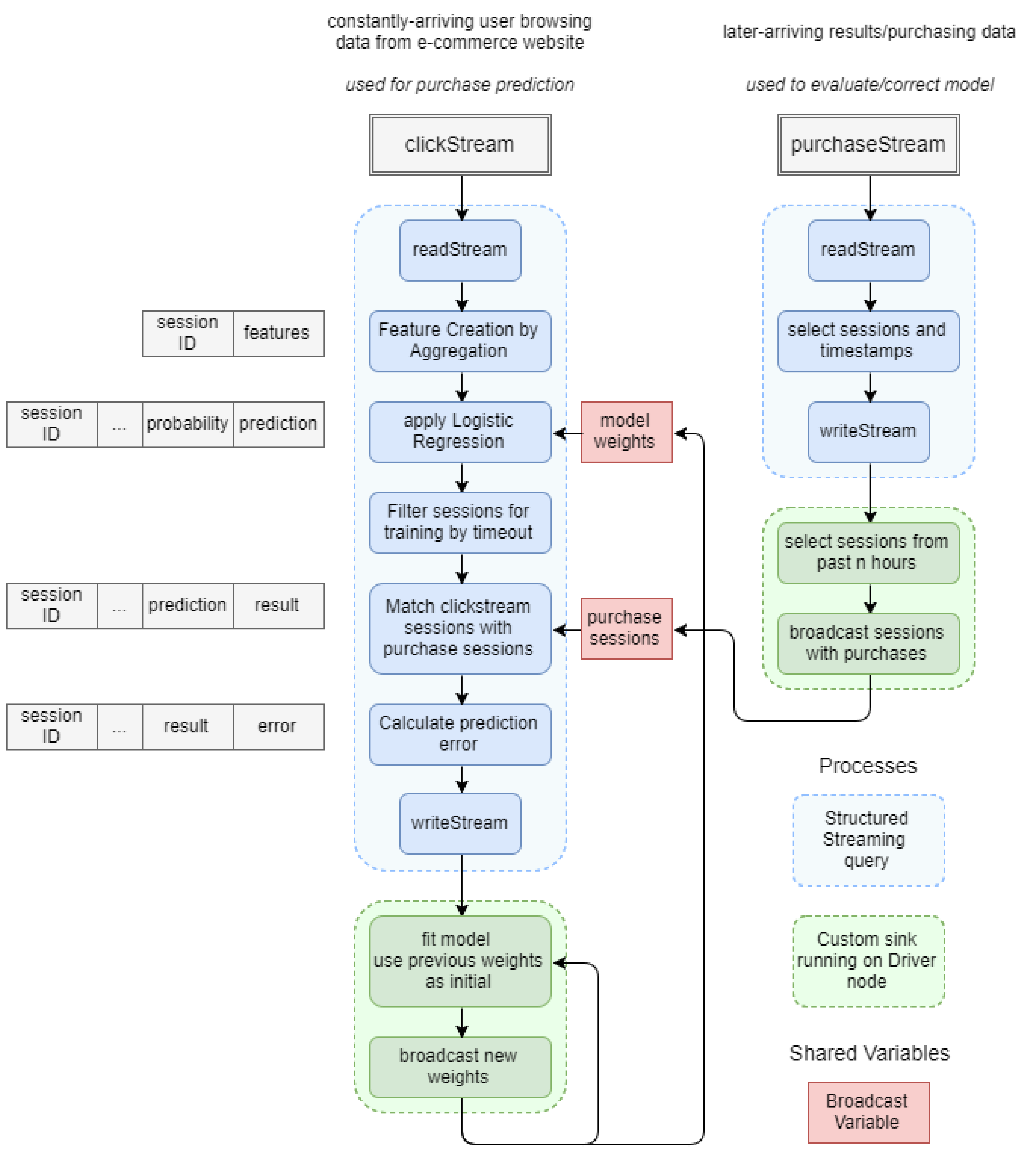
Continuously-adapting models can keep abreast of drifting environmental conditions, representing an improvement in mitigating model degradation over regular batch rebuilds.

Apache Spark

Apache Spark, along with its new Structured Streaming functionality, is a rapidly-growing distributed processing platform that could potentially allow continuous architectures within a single environment.

Scenario

This architecture has been developed using e-commerce user browsing and purchasing data from the 'RecSys Challenge 2015'. The imagined case study explored predicting the likelihood of a user making a purchase, perhaps to trigger offering threshold users a voucher to sway them.


constantly-arriving user browsing data from e-commerce website
used for purchase prediction
later-arriving results/purchasing data
used to evaluate/correct model
clickStream
purchaseStream
readStream
Feature Creation by Aggregation
apply Logistic Regression
Filter sessions for training by timeout
Match clickstream sessions with purchase sessions
Calculate prediction error
writeStream
fit model use previous weights as initial
broadcast new weights
model weights
purchase sessions
readStream
select sessions and timestamps
writeStream
select sessions from past n hours
broadcast sessions with purchases
session ID | features
session ID | ... | probability | prediction
session ID | ... | prediction | result
session ID | ... | result | error
Processes
Structured Streaming query
Custom sink running on Driver node
Shared Variables
Broadcast Variable

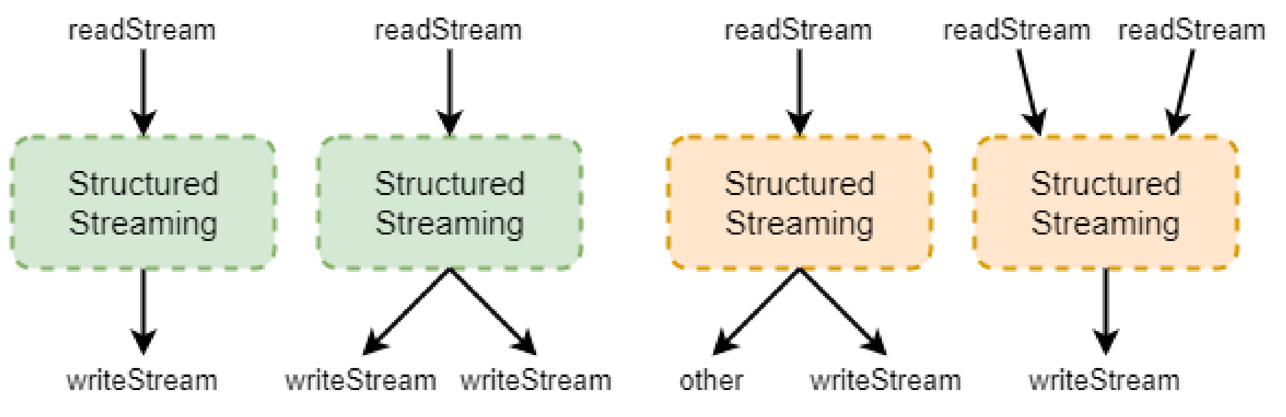
Challenges

Crossing the Streams

Structured Streaming queries have a rigidly defined and enclosed structure and *do not yet support stream-stream joins*. Moving data between streams can be achieved using *Broadcast Variables*, but triggering the broadcast of these *breaks query linearity* (see fig below, queries must end with a dataframe, and a writeStream).

Model Training

The new spark.ml machine learning library *does not yet have a streaming logistic regression model*, and a model transformer *cannot be re-trained* at each trigger of a Structured Streaming query.



Allowed (green) and disallowed (red) query structures

Solution

Solving both challenges could be possible by performing the model training and variable broadcast outside of the query and its restrictions by using a *custom sink*.

This could train a new model each time it is triggered, without the constraint of having to output a dataframe by writeStream.

Since this can be executed on the driver node it will have access to the SparkContext (necessary to broadcast variables), although this introduces a performance bottleneck.