package com.twitter.timelines.data\_processing.ml\_util.aggregation\_framework.conversion

import com.twitter.bijection.Injection

import com.twitter.ml.api.\_

import com.twitter.ml.api.util.SRichDataRecord

import com.twitter.scalding.TypedPipe

object DataSetPipeSketchJoin {

val DefaultSketchNumReducers = 500

val dataRecordMerger: DataRecordMerger = new DataRecordMerger

implicit val str2Byte: String => Array[Byte] =

implicitly[Injection[String, Array[Byte]]].toFunction

/\* Computes a left sketch join on a set of skewed keys. \*/

def apply(

inputDataSet: DataSetPipe,

skewedJoinKeys: Product,

joinFeaturesDataSet: DataSetPipe,

sketchNumReducers: Int = DefaultSketchNumReducers

): DataSetPipe = {

val joinKeyList = skewedJoinKeys.productIterator.toList.asInstanceOf[List[Feature[\_]]]

def makeKey(record: DataRecord): String =

joinKeyList

.map(SRichDataRecord(record).getFeatureValue(\_))

.toString

def byKey(pipe: DataSetPipe): TypedPipe[(String, DataRecord)] =

pipe.records.map(record => (makeKey(record), record))

val joinedRecords = byKey(inputDataSet)

.sketch(sketchNumReducers)

.leftJoin(byKey(joinFeaturesDataSet))

.values

.map {

case (inputRecord, joinFeaturesOpt) =>

joinFeaturesOpt.foreach { joinRecord => dataRecordMerger.merge(inputRecord, joinRecord) }

inputRecord

}

DataSetPipe(

joinedRecords,

FeatureContext.merge(inputDataSet.featureContext, joinFeaturesDataSet.featureContext)

)

}

}