package com.twitter.home\_mixer.product.scored\_tweets.feature\_hydrator.adapters.earlybird

import com.twitter.ml.api.Feature

import com.twitter.ml.api.FeatureContext

import com.twitter.ml.api.RichDataRecord

import com.twitter.ml.api.util.DataRecordConverters.RichDataRecordWrapper

import com.twitter.search.common.features.{thriftscala => sc}

import com.twitter.timelines.prediction.common.adapters.TimelinesMutatingAdapterBase

import com.twitter.timelines.prediction.features.common.InReplyToTweetTimelinesSharedFeatures

import com.twitter.timelines.prediction.features.recap.InReplyToRecapFeatures

import java.lang.{Boolean => JBoolean}

import java.lang.{Double => JDouble}

object InReplyToEarlybirdAdapter

extends TimelinesMutatingAdapterBase[Option[sc.ThriftTweetFeatures]] {

override val getFeatureContext: FeatureContext = new FeatureContext(

// TextFeatures

InReplyToTweetTimelinesSharedFeatures.WEIGHTED\_FAV\_COUNT,

InReplyToTweetTimelinesSharedFeatures.WEIGHTED\_RETWEET\_COUNT,

InReplyToTweetTimelinesSharedFeatures.WEIGHTED\_REPLY\_COUNT,

InReplyToTweetTimelinesSharedFeatures.WEIGHTED\_QUOTE\_COUNT,

InReplyToTweetTimelinesSharedFeatures.DECAYED\_FAVORITE\_COUNT,

InReplyToTweetTimelinesSharedFeatures.DECAYED\_RETWEET\_COUNT,

InReplyToTweetTimelinesSharedFeatures.DECAYED\_REPLY\_COUNT,

InReplyToTweetTimelinesSharedFeatures.DECAYED\_QUOTE\_COUNT,

InReplyToTweetTimelinesSharedFeatures.QUOTE\_COUNT,

InReplyToTweetTimelinesSharedFeatures.HAS\_QUOTE,

InReplyToTweetTimelinesSharedFeatures.EARLYBIRD\_SCORE,

InReplyToRecapFeatures.PREV\_USER\_TWEET\_ENGAGEMENT,

InReplyToRecapFeatures.IS\_SENSITIVE,

InReplyToRecapFeatures.IS\_AUTHOR\_NEW,

InReplyToRecapFeatures.NUM\_MENTIONS,

InReplyToRecapFeatures.HAS\_MENTION,

InReplyToRecapFeatures.HAS\_HASHTAG,

InReplyToRecapFeatures.IS\_AUTHOR\_NSFW,

InReplyToRecapFeatures.IS\_AUTHOR\_SPAM,

InReplyToRecapFeatures.IS\_AUTHOR\_BOT,

InReplyToRecapFeatures.FROM\_MUTUAL\_FOLLOW,

InReplyToRecapFeatures.USER\_REP,

InReplyToRecapFeatures.FROM\_VERIFIED\_ACCOUNT,

InReplyToRecapFeatures.HAS\_IMAGE,

InReplyToRecapFeatures.HAS\_NEWS,

InReplyToRecapFeatures.HAS\_VIDEO,

InReplyToRecapFeatures.HAS\_VISIBLE\_LINK,

InReplyToRecapFeatures.IS\_OFFENSIVE,

InReplyToRecapFeatures.IS\_REPLY,

InReplyToRecapFeatures.BIDIRECTIONAL\_REPLY\_COUNT,

InReplyToRecapFeatures.UNIDIRECTIONAL\_REPLY\_COUNT,

InReplyToRecapFeatures.BIDIRECTIONAL\_RETWEET\_COUNT,

InReplyToRecapFeatures.UNIDIRECTIONAL\_RETWEET\_COUNT,

InReplyToRecapFeatures.BIDIRECTIONAL\_FAV\_COUNT,

InReplyToRecapFeatures.UNIDIRECTIONAL\_FAV\_COUNT,

InReplyToRecapFeatures.CONVERSATIONAL\_COUNT,

InReplyToRecapFeatures.REPLY\_COUNT,

InReplyToRecapFeatures.RETWEET\_COUNT,

InReplyToRecapFeatures.FAV\_COUNT,

InReplyToRecapFeatures.TEXT\_SCORE,

InReplyToRecapFeatures.FAV\_COUNT\_V2,

InReplyToRecapFeatures.RETWEET\_COUNT\_V2,

InReplyToRecapFeatures.REPLY\_COUNT\_V2)

override val commonFeatures: Set[Feature[\_]] = Set.empty

override def setFeatures(

ebFeatures: Option[sc.ThriftTweetFeatures],

richDataRecord: RichDataRecord

): Unit = {

if (ebFeatures.nonEmpty) {

val features = ebFeatures.get

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.WEIGHTED\_FAV\_COUNT,

features.weightedFavoriteCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.WEIGHTED\_RETWEET\_COUNT,

features.weightedRetweetCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.WEIGHTED\_REPLY\_COUNT,

features.weightedReplyCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.WEIGHTED\_QUOTE\_COUNT,

features.weightedQuoteCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.DECAYED\_FAVORITE\_COUNT,

features.decayedFavoriteCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.DECAYED\_RETWEET\_COUNT,

features.decayedRetweetCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.DECAYED\_REPLY\_COUNT,

features.decayedReplyCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.DECAYED\_QUOTE\_COUNT,

features.decayedQuoteCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.QUOTE\_COUNT,

features.quoteCount.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToTweetTimelinesSharedFeatures.HAS\_QUOTE,

features.hasQuote

)

if (features.earlybirdScore > 0)

richDataRecord.setFeatureValue[JDouble](

InReplyToTweetTimelinesSharedFeatures.EARLYBIRD\_SCORE,

features.earlybirdScore

)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.PREV\_USER\_TWEET\_ENGAGEMENT,

features.prevUserTweetEngagement.toDouble

)

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.IS\_SENSITIVE, features.isSensitiveContent)

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.IS\_AUTHOR\_NEW, features.isAuthorNew)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.NUM\_MENTIONS,

features.numMentions.toDouble)

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.HAS\_MENTION, (features.numMentions > 0))

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.HAS\_HASHTAG, (features.numHashtags > 0))

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.IS\_AUTHOR\_NSFW, features.isAuthorNSFW)

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.IS\_AUTHOR\_SPAM, features.isAuthorSpam)

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.IS\_AUTHOR\_BOT, features.isAuthorBot)

richDataRecord.setFeatureValue[JBoolean](

InReplyToRecapFeatures.FROM\_MUTUAL\_FOLLOW,

features.fromMutualFollow)

richDataRecord.setFeatureValue[JDouble](InReplyToRecapFeatures.USER\_REP, features.userRep)

richDataRecord.setFeatureValue[JBoolean](

InReplyToRecapFeatures.FROM\_VERIFIED\_ACCOUNT,

features.fromVerifiedAccount)

richDataRecord.setFeatureValue[JBoolean](InReplyToRecapFeatures.HAS\_IMAGE, features.hasImage)

richDataRecord.setFeatureValue[JBoolean](InReplyToRecapFeatures.HAS\_NEWS, features.hasNews)

richDataRecord.setFeatureValue[JBoolean](InReplyToRecapFeatures.HAS\_VIDEO, features.hasVideo)

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.HAS\_VISIBLE\_LINK, features.hasVisibleLink)

richDataRecord

.setFeatureValue[JBoolean](InReplyToRecapFeatures.IS\_OFFENSIVE, features.isOffensive)

richDataRecord.setFeatureValue[JBoolean](InReplyToRecapFeatures.IS\_REPLY, features.isReply)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.BIDIRECTIONAL\_REPLY\_COUNT,

features.bidirectionalReplyCount)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.UNIDIRECTIONAL\_REPLY\_COUNT,

features.unidirectionalReplyCount)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.BIDIRECTIONAL\_RETWEET\_COUNT,

features.bidirectionalRetweetCount)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.UNIDIRECTIONAL\_RETWEET\_COUNT,

features.unidirectionalRetweetCount)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.BIDIRECTIONAL\_FAV\_COUNT,

features.bidirectionalFavCount)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.UNIDIRECTIONAL\_FAV\_COUNT,

features.unidirectionalFavCount)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.CONVERSATIONAL\_COUNT,

features.conversationCount)

richDataRecord

.setFeatureValue[JDouble](InReplyToRecapFeatures.REPLY\_COUNT, features.replyCount.toDouble)

richDataRecord.setFeatureValue[JDouble](

InReplyToRecapFeatures.RETWEET\_COUNT,

features.retweetCount.toDouble)

richDataRecord

.setFeatureValue[JDouble](InReplyToRecapFeatures.FAV\_COUNT, features.favCount.toDouble)

richDataRecord.setFeatureValue[JDouble](InReplyToRecapFeatures.TEXT\_SCORE, features.textScore)

richDataRecord.setFeatureValueFromOption(

InReplyToRecapFeatures.FAV\_COUNT\_V2,

features.favCountV2.map(\_.toDouble))

richDataRecord.setFeatureValueFromOption(

InReplyToRecapFeatures.RETWEET\_COUNT\_V2,

features.retweetCountV2.map(\_.toDouble)

)

richDataRecord.setFeatureValueFromOption(

InReplyToRecapFeatures.REPLY\_COUNT\_V2,

features.replyCountV2.map(\_.toDouble))

}

}

}