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

import com.twitter.home\_mixer.model.HomeFeatures.CandidateSourceIdFeature

import com.twitter.home\_mixer.util.CandidatesUtil

import com.twitter.ml.api.DataRecord

import com.twitter.ml.api.RichDataRecord

import com.twitter.ml.api.constant.SharedFeatures

import com.twitter.ml.api.util.DataRecordConverters.\_

import com.twitter.product\_mixer.component\_library.model.candidate.TweetCandidate

import com.twitter.product\_mixer.core.feature.Feature

import com.twitter.product\_mixer.core.feature.FeatureWithDefaultOnFailure

import com.twitter.product\_mixer.core.feature.datarecord.DataRecordInAFeature

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMapBuilder

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.CandidateFeatureHydrator

import com.twitter.product\_mixer.core.model.common.identifier.FeatureHydratorIdentifier

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.product\_mixer.core.util.OffloadFuturePools

import com.twitter.stitch.Stitch

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

import java.lang.{Long => JLong}

object TweetMetaDataDataRecord

extends DataRecordInAFeature[TweetCandidate]

with FeatureWithDefaultOnFailure[TweetCandidate, DataRecord] {

override def defaultValue: DataRecord = new DataRecord()

}

object TweetMetaDataFeatureHydrator

extends CandidateFeatureHydrator[PipelineQuery, TweetCandidate] {

override val identifier: FeatureHydratorIdentifier = FeatureHydratorIdentifier("TweetMetaData")

override def features: Set[Feature[\_, \_]] = Set(TweetMetaDataDataRecord)

override def apply(

query: PipelineQuery,

candidate: TweetCandidate,

existingFeatures: FeatureMap

): Stitch[FeatureMap] = OffloadFuturePools.offload {

val richDataRecord = new RichDataRecord()

setFeatures(richDataRecord, candidate, existingFeatures)

FeatureMapBuilder().add(TweetMetaDataDataRecord, richDataRecord.getRecord).build()

}

private def setFeatures(

richDataRecord: RichDataRecord,

candidate: TweetCandidate,

existingFeatures: FeatureMap

): Unit = {

richDataRecord.setFeatureValue[JLong](SharedFeatures.TWEET\_ID, candidate.id)

richDataRecord.setFeatureValueFromOption(

TimelinesSharedFeatures.ORIGINAL\_AUTHOR\_ID,

CandidatesUtil.getOriginalAuthorId(existingFeatures))

richDataRecord.setFeatureValueFromOption(

TimelinesSharedFeatures.CANDIDATE\_TWEET\_SOURCE\_ID,

existingFeatures.getOrElse(CandidateSourceIdFeature, None).map(\_.value.toLong))

}

}