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

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

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

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

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

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

import com.twitter.ml.api.DataRecord

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.adapters.real\_graph.RealGraphEdgeFeaturesCombineAdapter

import com.twitter.timelines.real\_graph.v1.{thriftscala => v1}

import javax.inject.Inject

import javax.inject.Singleton

import scala.collection.JavaConverters.\_

object RealGraphViewerRelatedUsersDataRecordFeature

extends DataRecordInAFeature[TweetCandidate]

with FeatureWithDefaultOnFailure[TweetCandidate, DataRecord] {

override def defaultValue: DataRecord = new DataRecord()

}

@Singleton

class RealGraphViewerRelatedUsersFeatureHydrator @Inject() ()

extends CandidateFeatureHydrator[PipelineQuery, TweetCandidate] {

override val identifier: FeatureHydratorIdentifier =

FeatureHydratorIdentifier("RealGraphViewerRelatedUsers")

override val features: Set[Feature[\_, \_]] = Set(RealGraphViewerRelatedUsersDataRecordFeature)

private val RealGraphEdgeFeaturesCombineAdapter = new RealGraphEdgeFeaturesCombineAdapter

override def apply(

query: PipelineQuery,

candidate: TweetCandidate,

existingFeatures: FeatureMap

): Stitch[FeatureMap] = OffloadFuturePools.offload {

val realGraphQueryFeatures = query.features

.flatMap(\_.getOrElse(RealGraphFeatures, None))

.getOrElse(Map.empty[Long, v1.RealGraphEdgeFeatures])

val allRelatedUserIds = getRelatedUserIds(existingFeatures)

val realGraphFeatures = RealGraphViewerAuthorFeatureHydrator.getCombinedRealGraphFeatures(

allRelatedUserIds,

realGraphQueryFeatures

)

val realGraphFeaturesDataRecord = RealGraphEdgeFeaturesCombineAdapter

.adaptToDataRecords(Some(realGraphFeatures)).asScala.headOption

.getOrElse(new DataRecord)

FeatureMapBuilder()

.add(RealGraphViewerRelatedUsersDataRecordFeature, realGraphFeaturesDataRecord)

.build()

}

private def getRelatedUserIds(features: FeatureMap): Seq[Long] = {

(CandidatesUtil.getEngagerUserIds(features) ++

features.getOrElse(AuthorIdFeature, None) ++

features.getOrElse(MentionUserIdFeature, Seq.empty) ++

features.getOrElse(SourceUserIdFeature, None) ++

features.getOrElse(DirectedAtUserIdFeature, None)).distinct

}

}