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

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

import com.twitter.product\_mixer.component\_library.candidate\_source.timeline\_ranker.TimelineRankerInNetworkSourceTweetsByTweetIdMapFeature

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.featuremap.FeatureMap

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

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

import com.twitter.product\_mixer.core.model.common.CandidateWithFeatures

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

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

import com.twitter.search.common.features.thriftscala.ThriftTweetFeatures

import com.twitter.stitch.Stitch

import com.twitter.timelineranker.thriftscala.CandidateTweet

object SourceTweetEarlybirdFeature extends Feature[TweetCandidate, Option[ThriftTweetFeatures]]

/\*\*

\* Feature Hydrator that bulk hydrates source tweets' features to retweet candidates

\*/

object RetweetSourceTweetFeatureHydrator

extends BulkCandidateFeatureHydrator[PipelineQuery, TweetCandidate] {

override val identifier: FeatureHydratorIdentifier =

FeatureHydratorIdentifier("RetweetSourceTweet")

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

SourceTweetEarlybirdFeature,

)

private val DefaultFeatureMap = FeatureMapBuilder()

.add(SourceTweetEarlybirdFeature, None)

.build()

override def apply(

query: PipelineQuery,

candidates: Seq[CandidateWithFeatures[TweetCandidate]]

): Stitch[Seq[FeatureMap]] = {

val sourceTweetsByTweetId: Option[Map[Long, CandidateTweet]] = {

query.features.map(

\_.getOrElse(

TimelineRankerInNetworkSourceTweetsByTweetIdMapFeature,

Map.empty[Long, CandidateTweet]))

}

/\*\*

\* Return DefaultFeatureMap (no-op to candidate) when it is unfeasible to hydrate the

\* source tweet's feature to the current candidate: early bird does not return source

\* tweets info / candidate is not a retweet / sourceTweetId is not found

\*/

Stitch.value {

if (sourceTweetsByTweetId.exists(\_.nonEmpty)) {

candidates.map { candidate =>

val candidateIsRetweet = candidate.features.getOrElse(IsRetweetFeature, false)

val sourceTweetId = candidate.features.getOrElse(SourceTweetIdFeature, None)

if (!candidateIsRetweet || sourceTweetId.isEmpty) {

DefaultFeatureMap

} else {

val sourceTweet = sourceTweetsByTweetId.flatMap(\_.get(sourceTweetId.get))

if (sourceTweet.nonEmpty) {

val source = sourceTweet.get

FeatureMapBuilder()

.add(SourceTweetEarlybirdFeature, source.features)

.build()

} else {

DefaultFeatureMap

}

}

}

} else {

candidates.map(\_ => DefaultFeatureMap)

}

}

}

}