package com.twitter.home\_mixer.product.scored\_tweets.query\_transformer

import com.twitter.conversions.DurationOps.\_

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

import com.twitter.home\_mixer.model.request.HasDeviceContext

import com.twitter.home\_mixer.product.scored\_tweets.param.ScoredTweetsParam

import com.twitter.home\_mixer.product.scored\_tweets.query\_transformer.TimelineRankerUtegQueryTransformer.\_

import com.twitter.home\_mixer.util.earlybird.EarlybirdRequestUtil

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidatePipelineQueryTransformer

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

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

import com.twitter.product\_mixer.core.quality\_factor.HasQualityFactorStatus

import com.twitter.timelineranker.{model => tlr}

import com.twitter.timelineranker.{thriftscala => t}

import com.twitter.timelines.common.model.TweetKindOption

import com.twitter.timelines.earlybird.common.options.EarlybirdScoringModelConfig

import com.twitter.timelines.model.UserId

import com.twitter.timelines.model.candidate.CandidateTweetSourceId

object TimelineRankerUtegQueryTransformer {

private val SinceDuration = 24.hours

private val MaxTweetsToFetch = 300

private val MaxUtegCandidates = 800

private val tweetKindOptions =

TweetKindOption(includeOriginalTweetsAndQuotes = true, includeReplies = true)

def utegEarlybirdModels: Seq[EarlybirdScoringModelConfig] =

EarlybirdRequestUtil.EarlybirdScoringModels.UnifiedEngagementRectweet

}

case class TimelineRankerUtegQueryTransformer[

Query <: PipelineQuery with HasQualityFactorStatus with HasDeviceContext

](

override val candidatePipelineIdentifier: CandidatePipelineIdentifier,

override val maxTweetsToFetch: Int = MaxTweetsToFetch)

extends CandidatePipelineQueryTransformer[Query, t.UtegLikedByTweetsQuery]

with TimelineRankerQueryTransformer[Query] {

override val candidateTweetSourceId = CandidateTweetSourceId.RecommendedTweet

override val options = tweetKindOptions

override val earlybirdModels = utegEarlybirdModels

override def getTensorflowModel(query: Query): Option[String] = {

Some(query.params(ScoredTweetsParam.EarlybirdTensorflowModel.UtegParam))

}

override def utegLikedByTweetsOptions(input: Query): Option[tlr.UtegLikedByTweetsOptions] = Some(

tlr.UtegLikedByTweetsOptions(

utegCount = MaxUtegCandidates,

isInNetwork = false,

weightedFollowings = input.features

.map(\_.getOrElse(RealGraphInNetworkScoresFeature, Map.empty[UserId, Double]))

.getOrElse(Map.empty)

)

)

override def transform(input: Query): t.UtegLikedByTweetsQuery =

buildTimelineRankerQuery(input, SinceDuration).toThriftUtegLikedByTweetsQuery

}