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

import com.twitter.finagle.stats.StatsReceiver

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

import com.twitter.home\_mixer.param.HomeMixerInjectionNames.TopicEngagementCache

import com.twitter.ml.api.DataRecord

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

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

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

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.servo.cache.ReadCache

import com.twitter.timelines.data\_processing.ml\_util.aggregation\_framework.AggregateGroup

import com.twitter.timelines.prediction.common.aggregates.real\_time.TimelinesOnlineAggregationFeaturesOnlyConfig.\_

import javax.inject.Inject

import javax.inject.Named

import javax.inject.Singleton

object TopicEngagementRealTimeAggregateFeature

extends DataRecordInAFeature[TweetCandidate]

with FeatureWithDefaultOnFailure[TweetCandidate, DataRecord] {

override def defaultValue: DataRecord = new DataRecord()

}

@Singleton

class TopicEngagementRealTimeAggregateFeatureHydrator @Inject() (

@Named(TopicEngagementCache) override val client: ReadCache[Long, DataRecord],

override val statsReceiver: StatsReceiver)

extends BaseRealTimeAggregateBulkCandidateFeatureHydrator[Long] {

override val identifier: FeatureHydratorIdentifier =

FeatureHydratorIdentifier("TopicEngagementRealTimeAggregate")

override val outputFeature: DataRecordInAFeature[TweetCandidate] =

TopicEngagementRealTimeAggregateFeature

override val aggregateGroups: Seq[AggregateGroup] = Seq(

topicEngagementRealTimeAggregatesProd,

topicEngagement24HourRealTimeAggregatesProd,

topicShareEngagementsRealTimeAggregates

)

override val aggregateGroupToPrefix: Map[AggregateGroup, String] = Map(

topicEngagement24HourRealTimeAggregatesProd -> "topic.timelines.topic\_engagement\_24\_hour\_real\_time\_aggregates.",

topicShareEngagementsRealTimeAggregates -> "topic.timelines.topic\_share\_engagements\_real\_time\_aggregates."

)

override def keysFromQueryAndCandidates(

query: PipelineQuery,

candidates: Seq[CandidateWithFeatures[TweetCandidate]]

): Seq[Option[Long]] = {

candidates.map { candidate =>

candidate.features

.getTry(TopicIdSocialContextFeature)

.toOption

.flatten

}

}

}