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

import com.google.inject.name.Named

import com.twitter.finagle.stats.StatsReceiver

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

import com.twitter.ml.api.DataRecord

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.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.Singleton

object UserEngagementRealTimeAggregateFeature

extends DataRecordInAFeature[PipelineQuery]

with FeatureWithDefaultOnFailure[PipelineQuery, DataRecord] {

override def defaultValue: DataRecord = new DataRecord()

}

@Singleton

class UserEngagementRealTimeAggregatesFeatureHydrator @Inject() (

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

override val statsReceiver: StatsReceiver)

extends BaseRealTimeAggregateQueryFeatureHydrator[Long] {

override val identifier: FeatureHydratorIdentifier =

FeatureHydratorIdentifier("UserEngagementRealTimeAggregates")

override val outputFeature: DataRecordInAFeature[PipelineQuery] =

UserEngagementRealTimeAggregateFeature

val aggregateGroups: Seq[AggregateGroup] = Seq(

userEngagementRealTimeAggregatesProd,

userShareEngagementsRealTimeAggregates,

userBCEDwellEngagementsRealTimeAggregates,

userEngagement48HourRealTimeAggregatesProd,

userNegativeEngagementAuthorUserState72HourRealTimeAggregates,

userNegativeEngagementAuthorUserStateRealTimeAggregates,

userProfileEngagementRealTimeAggregates,

)

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

userShareEngagementsRealTimeAggregates -> "user.timelines.user\_share\_engagements\_real\_time\_aggregates.",

userBCEDwellEngagementsRealTimeAggregates -> "user.timelines.user\_bce\_dwell\_engagements\_real\_time\_aggregates.",

userEngagement48HourRealTimeAggregatesProd -> "user.timelines.user\_engagement\_48\_hour\_real\_time\_aggregates.",

userNegativeEngagementAuthorUserState72HourRealTimeAggregates -> "user.timelines.user\_negative\_engagement\_author\_user\_state\_72\_hour\_real\_time\_aggregates.",

userProfileEngagementRealTimeAggregates -> "user.timelines.user\_profile\_engagement\_real\_time\_aggregates."

)

override def keysFromQueryAndCandidates(query: PipelineQuery): Option[Long] = {

Some(query.getRequiredUserId)

}

}