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

import com.twitter.conversions.DurationOps.\_

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

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

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

import com.twitter.ml.api.DataRecord

import com.twitter.ml.api.util.FDsl.\_

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.QueryFeatureHydrator

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

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

import com.twitter.snowflake.id.SnowflakeId

import com.twitter.stitch.Stitch

import com.twitter.timelines.prediction.features.time\_features.AccountAgeInterval

import com.twitter.timelines.prediction.features.time\_features.TimeDataRecordFeatures.ACCOUNT\_AGE\_INTERVAL

import com.twitter.timelines.prediction.features.time\_features.TimeDataRecordFeatures.IS\_12\_MONTH\_NEW\_USER

import com.twitter.timelines.prediction.features.time\_features.TimeDataRecordFeatures.IS\_30\_DAY\_NEW\_USER

import com.twitter.timelines.prediction.features.time\_features.TimeDataRecordFeatures.TIME\_BETWEEN\_NON\_POLLING\_REQUESTS\_AVG

import com.twitter.timelines.prediction.features.time\_features.TimeDataRecordFeatures.TIME\_SINCE\_LAST\_NON\_POLLING\_REQUEST

import com.twitter.timelines.prediction.features.time\_features.TimeDataRecordFeatures.TIME\_SINCE\_VIEWER\_ACCOUNT\_CREATION\_SECS

import com.twitter.timelines.prediction.features.time\_features.TimeDataRecordFeatures.USER\_ID\_IS\_SNOWFLAKE\_ID

import com.twitter.user\_session\_store.ReadRequest

import com.twitter.user\_session\_store.ReadWriteUserSessionStore

import com.twitter.user\_session\_store.UserSessionDataset

import com.twitter.user\_session\_store.UserSessionDataset.UserSessionDataset

import com.twitter.util.Time

import javax.inject.Inject

import javax.inject.Singleton

object RequestTimeDataRecordFeature

extends DataRecordInAFeature[PipelineQuery]

with FeatureWithDefaultOnFailure[PipelineQuery, DataRecord] {

override def defaultValue: DataRecord = new DataRecord()

}

@Singleton

case class RequestTimeQueryFeatureHydrator @Inject() (

userSessionStore: ReadWriteUserSessionStore)

extends QueryFeatureHydrator[PipelineQuery] {

override val identifier: FeatureHydratorIdentifier = FeatureHydratorIdentifier("RequestTime")

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

FollowingLastNonPollingTimeFeature,

LastNonPollingTimeFeature,

NonPollingTimesFeature,

RequestTimeDataRecordFeature

)

private val datasets: Set[UserSessionDataset] = Set(UserSessionDataset.NonPollingTimes)

override def hydrate(query: PipelineQuery): Stitch[FeatureMap] = {

userSessionStore

.read(ReadRequest(query.getRequiredUserId, datasets))

.map { userSession =>

val nonPollingTimestamps = userSession.flatMap(\_.nonPollingTimestamps)

val lastNonPollingTime = nonPollingTimestamps

.flatMap(\_.nonPollingTimestampsMs.headOption)

.map(Time.fromMilliseconds)

val followingLastNonPollingTime = nonPollingTimestamps

.flatMap(\_.mostRecentHomeLatestNonPollingTimestampMs)

.map(Time.fromMilliseconds)

val nonPollingTimes = nonPollingTimestamps

.map(\_.nonPollingTimestampsMs)

.getOrElse(Seq.empty)

val requestTimeDataRecord = getRequestTimeDataRecord(query, nonPollingTimes)

FeatureMapBuilder()

.add(FollowingLastNonPollingTimeFeature, followingLastNonPollingTime)

.add(LastNonPollingTimeFeature, lastNonPollingTime)

.add(NonPollingTimesFeature, nonPollingTimes)

.add(RequestTimeDataRecordFeature, requestTimeDataRecord)

.build()

}

}

def getRequestTimeDataRecord(query: PipelineQuery, nonPollingTimes: Seq[Long]): DataRecord = {

val requestTimeMs = query.queryTime.inMillis

val accountAge = SnowflakeId.timeFromIdOpt(query.getRequiredUserId)

val timeSinceAccountCreation = accountAge.map(query.queryTime.since)

val timeSinceEarliestNonPollingRequest =

nonPollingTimes.lastOption.map(requestTimeMs - \_)

val timeSinceLastNonPollingRequest =

nonPollingTimes.headOption.map(requestTimeMs - \_)

new DataRecord()

.setFeatureValue(USER\_ID\_IS\_SNOWFLAKE\_ID, accountAge.isDefined)

.setFeatureValue(

IS\_30\_DAY\_NEW\_USER,

timeSinceAccountCreation.map(\_ < 30.days).getOrElse(false)

)

.setFeatureValue(

IS\_12\_MONTH\_NEW\_USER,

timeSinceAccountCreation.map(\_ < 365.days).getOrElse(false)

)

.setFeatureValueFromOption(

ACCOUNT\_AGE\_INTERVAL,

timeSinceAccountCreation.flatMap(AccountAgeInterval.fromDuration).map(\_.id.toLong)

)

.setFeatureValueFromOption(

TIME\_SINCE\_VIEWER\_ACCOUNT\_CREATION\_SECS,

timeSinceAccountCreation.map(\_.inSeconds.toDouble)

)

.setFeatureValueFromOption(

TIME\_BETWEEN\_NON\_POLLING\_REQUESTS\_AVG,

timeSinceEarliestNonPollingRequest.map(\_.toDouble / math.max(1.0, nonPollingTimes.size))

)

.setFeatureValueFromOption(

TIME\_SINCE\_LAST\_NON\_POLLING\_REQUEST,

timeSinceLastNonPollingRequest.map(\_.toDouble)

)

}

}