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

import com.twitter.ml.api.FeatureContext

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

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

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

import scala.jdk.CollectionConverters.asJavaIterableConverter

// A helper class deriving aggregate feature info from the given configuration parameters.

class AggregateFeatureInfo(

val aggregateGroups: Set[AggregateGroup],

val aggregateType: AggregateType) {

private val typedAggregateGroups = aggregateGroups.flatMap(\_.buildTypedAggregateGroups()).toList

val featureContext: FeatureContext =

new FeatureContext(

(typedAggregateGroups.flatMap(\_.allOutputFeatures) ++

typedAggregateGroups.flatMap(\_.allOutputKeys) ++

Seq(TypedAggregateGroup.timestampFeature)).asJava)

val feature: BaseAggregateRootFeature =

AggregateFeatureInfo.pickFeature(aggregateType)

}

object AggregateFeatureInfo {

val features: Set[BaseAggregateRootFeature] =

Set(PartAAggregateRootFeature, PartBAggregateRootFeature)

def pickFeature(aggregateType: AggregateType): BaseAggregateRootFeature = {

val filtered = features.filter(\_.aggregateTypes.contains(aggregateType))

require(

filtered.size == 1,

"requested AggregateType must be backed by exactly one physical store.")

filtered.head

}

}