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

import com.twitter.home\_mixer.product.scored\_tweets.feature\_hydrator.adapters.offline\_aggregates.PassThroughAdapter

import com.twitter.home\_mixer.product.scored\_tweets.feature\_hydrator.adapters.offline\_aggregates.SparseAggregatesToDenseAdapter

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

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

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

import com.twitter.home\_mixer.product.scored\_tweets.feature\_hydrator.TSPInferredTopicFeature

import com.twitter.home\_mixer.util.CandidatesUtil

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

import com.twitter.timelines.prediction.common.aggregates.TimelinesAggregationConfig

import com.twitter.timelines.prediction.common.aggregates.TimelinesAggregationConfig.CombineCountPolicies

object EdgeAggregateFeatures {

object UserAuthorAggregateFeature

extends BaseEdgeAggregateFeature(

aggregateGroups = TimelinesAggregationConfig.userAuthorAggregatesV2 ++ Set(

TimelinesAggregationConfig.userAuthorAggregatesV5,

TimelinesAggregationConfig.tweetSourceUserAuthorAggregatesV1,

TimelinesAggregationConfig.twitterWideUserAuthorAggregates

),

aggregateType = AggregateType.UserAuthor,

extractMapFn = \_.userAuthorAggregates,

adapter = PassThroughAdapter,

getSecondaryKeysFn = \_.features.getOrElse(AuthorIdFeature, None).toSeq

)

object UserOriginalAuthorAggregateFeature

extends BaseEdgeAggregateFeature(

aggregateGroups = Set(TimelinesAggregationConfig.userOriginalAuthorAggregatesV1),

aggregateType = AggregateType.UserOriginalAuthor,

extractMapFn = \_.userOriginalAuthorAggregates,

adapter = PassThroughAdapter,

getSecondaryKeysFn = candidate =>

CandidatesUtil.getOriginalAuthorId(candidate.features).toSeq

)

object UserTopicAggregateFeature

extends BaseEdgeAggregateFeature(

aggregateGroups = Set(

TimelinesAggregationConfig.userTopicAggregates,

TimelinesAggregationConfig.userTopicAggregatesV2,

),

aggregateType = AggregateType.UserTopic,

extractMapFn = \_.userTopicAggregates,

adapter = PassThroughAdapter,

getSecondaryKeysFn = candidate =>

candidate.features.getOrElse(TopicIdSocialContextFeature, None).toSeq

)

object UserMentionAggregateFeature

extends BaseEdgeAggregateFeature(

aggregateGroups = Set(TimelinesAggregationConfig.userMentionAggregates),

aggregateType = AggregateType.UserMention,

extractMapFn = \_.userMentionAggregates,

adapter = new SparseAggregatesToDenseAdapter(CombineCountPolicies.MentionCountsPolicy),

getSecondaryKeysFn = candidate =>

candidate.features.getOrElse(MentionScreenNameFeature, Seq.empty).map(\_.hashCode.toLong)

)

object UserInferredTopicAggregateFeature

extends BaseEdgeAggregateFeature(

aggregateGroups = Set(

TimelinesAggregationConfig.userInferredTopicAggregates,

),

aggregateType = AggregateType.UserInferredTopic,

extractMapFn = \_.userInferredTopicAggregates,

adapter = new SparseAggregatesToDenseAdapter(

CombineCountPolicies.UserInferredTopicCountsPolicy),

getSecondaryKeysFn = candidate =>

candidate.features.getOrElse(TSPInferredTopicFeature, Map.empty[Long, Double]).keys.toSeq

)

object UserInferredTopicAggregateV2Feature

extends BaseEdgeAggregateFeature(

aggregateGroups = Set(

TimelinesAggregationConfig.userInferredTopicAggregatesV2

),

aggregateType = AggregateType.UserInferredTopic,

extractMapFn = \_.userInferredTopicAggregates,

adapter = new SparseAggregatesToDenseAdapter(

CombineCountPolicies.UserInferredTopicV2CountsPolicy),

getSecondaryKeysFn = candidate =>

candidate.features.getOrElse(TSPInferredTopicFeature, Map.empty[Long, Double]).keys.toSeq

)

object UserMediaUnderstandingAnnotationAggregateFeature

extends BaseEdgeAggregateFeature(

aggregateGroups = Set(

TimelinesAggregationConfig.userMediaUnderstandingAnnotationAggregates),

aggregateType = AggregateType.UserMediaUnderstandingAnnotation,

extractMapFn = \_.userMediaUnderstandingAnnotationAggregates,

adapter = new SparseAggregatesToDenseAdapter(

CombineCountPolicies.UserMediaUnderstandingAnnotationCountsPolicy),

getSecondaryKeysFn = candidate =>

CandidatesUtil.getMediaUnderstandingAnnotationIds(candidate.features)

)

object UserEngagerAggregateFeature

extends BaseEdgeAggregateFeature(

aggregateGroups = Set(TimelinesAggregationConfig.userEngagerAggregates),

aggregateType = AggregateType.UserEngager,

extractMapFn = \_.userEngagerAggregates,

adapter = new SparseAggregatesToDenseAdapter(CombineCountPolicies.EngagerCountsPolicy),

getSecondaryKeysFn = candidate => CandidatesUtil.getEngagerUserIds(candidate.features)

)

object UserEngagerGoodClickAggregateFeature

extends BaseEdgeAggregateFeature(

aggregateGroups = Set(TimelinesAggregationConfig.userEngagerGoodClickAggregates),

aggregateType = AggregateType.UserEngager,

extractMapFn = \_.userEngagerAggregates,

adapter = new SparseAggregatesToDenseAdapter(

CombineCountPolicies.EngagerGoodClickCountsPolicy),

getSecondaryKeysFn = candidate => CandidatesUtil.getEngagerUserIds(candidate.features)

)

}