package com.twitter.timelines.prediction.common.aggregates

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

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

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

trait TimelinesAggregationConfigTrait

extends TimelinesAggregationConfigDetails

with AggregationConfig {

private val aggregateGroups = Set(

authorTopicAggregates,

userTopicAggregates,

userTopicAggregatesV2,

userInferredTopicAggregates,

userInferredTopicAggregatesV2,

userAggregatesV2,

userAggregatesV5Continuous,

userReciprocalEngagementAggregates,

userAuthorAggregatesV5,

userOriginalAuthorReciprocalEngagementAggregates,

originalAuthorReciprocalEngagementAggregates,

tweetSourceUserAuthorAggregatesV1,

userEngagerAggregates,

userMentionAggregates,

twitterWideUserAggregates,

twitterWideUserAuthorAggregates,

userRequestHourAggregates,

userRequestDowAggregates,

userListAggregates,

userMediaUnderstandingAnnotationAggregates,

) ++ userAuthorAggregatesV2

val aggregatesToComputeList: Set[List[TypedAggregateGroup[\_]]] =

aggregateGroups.map(\_.buildTypedAggregateGroups())

override val aggregatesToCompute: Set[TypedAggregateGroup[\_]] = aggregatesToComputeList.flatten

/\*

\* Feature selection config to save storage space and manhattan query bandwidth.

\* Only the most important features found using offline RCE simulations are used

\* when actually training and serving. This selector is used by

\* [[com.twitter.timelines.data\_processing.jobs.timeline\_ranking\_user\_features.TimelineRankingAggregatesV2FeaturesProdJob]]

\* but defined here to keep it in sync with the config that computes the aggregates.

\*/

val AggregatesV2FeatureSelector = FeatureSelectorConfig.AggregatesV2ProdFeatureSelector

def filterAggregatesGroups(storeNames: Set[String]): Set[AggregateGroup] = {

aggregateGroups.filter(aggregateGroup => storeNames.contains(aggregateGroup.outputStore.name))

}

}