package com.twitter.follow\_recommendations.common.feature\_hydration.sources

import com.twitter.ml.api.DataRecord

import com.twitter.ml.api.FeatureContext

import com.twitter.ml.featurestore.catalog.entities.core.{Author => AuthorEntity}

import com.twitter.ml.featurestore.catalog.entities.core.{AuthorTopic => AuthorTopicEntity}

import com.twitter.ml.featurestore.catalog.entities.core.{CandidateUser => CandidateUserEntity}

import com.twitter.ml.featurestore.catalog.entities.core.{Topic => TopicEntity}

import com.twitter.ml.featurestore.catalog.entities.core.{User => UserEntity}

import com.twitter.ml.featurestore.catalog.entities.core.{UserCandidate => UserCandidateEntity}

import com.twitter.ml.featurestore.catalog.entities.onboarding.UserWtfAlgorithmEntity

import com.twitter.ml.featurestore.catalog.entities.onboarding.{

WtfAlgorithm => WtfAlgorithmIdEntity

}

import com.twitter.ml.featurestore.catalog.entities.onboarding.{

WtfAlgorithmType => WtfAlgorithmTypeEntity

}

import com.twitter.ml.featurestore.catalog.features.core.UserClients.FullPrimaryClientVersion

import com.twitter.ml.featurestore.catalog.features.core.UserClients.NumClients

import com.twitter.ml.featurestore.catalog.features.core.UserClients.PrimaryClient

import com.twitter.ml.featurestore.catalog.features.core.UserClients.PrimaryClientVersion

import com.twitter.ml.featurestore.catalog.features.core.UserClients.PrimaryDeviceManufacturer

import com.twitter.ml.featurestore.catalog.features.core.UserClients.PrimaryMobileSdkVersion

import com.twitter.ml.featurestore.catalog.features.core.UserClients.SecondaryClient

import com.twitter.ml.featurestore.catalog.features.core.UserCounts.Favorites

import com.twitter.ml.featurestore.catalog.features.core.UserCounts.Followers

import com.twitter.ml.featurestore.catalog.features.core.UserCounts.Following

import com.twitter.ml.featurestore.catalog.features.core.UserCounts.Tweets

import com.twitter.ml.featurestore.catalog.features.customer\_journey.PostNuxAlgorithmIdAggregateFeatureGroup

import com.twitter.ml.featurestore.catalog.features.customer\_journey.PostNuxAlgorithmTypeAggregateFeatureGroup

import com.twitter.ml.featurestore.catalog.features.customer\_journey.{Utils => FeatureGroupUtils}

import com.twitter.ml.featurestore.catalog.features.interests\_discovery.UserTopicRelationships.FollowedTopics

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumFavorites

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumFavoritesReceived

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumFollowBacks

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumFollows

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumFollowsReceived

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumLoginDays

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumLoginTweetImpressions

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumMuteBacks

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumMuted

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumOriginalTweets

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumQualityFollowReceived

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumQuoteRetweets

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumQuoteRetweetsReceived

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumReplies

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumRepliesReceived

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumRetweets

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumRetweetsReceived

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumSpamBlocked

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumSpamBlockedBacks

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumTweetImpressions

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumTweets

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumUnfollowBacks

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumUnfollows

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumUserActiveMinutes

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumWasMutualFollowed

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumWasMutualUnfollowed

import com.twitter.ml.featurestore.catalog.features.onboarding.MetricCenterUserCounts.NumWasUnfollowed

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.Country

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.FollowersOverFollowingRatio

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.Language

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.MutualFollowsOverFollowersRatio

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.MutualFollowsOverFollowingRatio

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.NumFollowers

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.NumFollowings

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.NumMutualFollows

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.TweepCred

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOffline.UserState

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOfflineEdge.HaveSameCountry

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOfflineEdge.HaveSameLanguage

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOfflineEdge.HaveSameUserState

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOfflineEdge.NumFollowersGap

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOfflineEdge.NumFollowingsGap

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOfflineEdge.NumMutualFollowsGap

import com.twitter.ml.featurestore.catalog.features.onboarding.PostNuxOfflineEdge.TweepCredGap

import com.twitter.ml.featurestore.catalog.features.onboarding.Ratio.FollowersFollowings

import com.twitter.ml.featurestore.catalog.features.onboarding.Ratio.MutualFollowsFollowing

import com.twitter.ml.featurestore.catalog.features.onboarding.SimclusterUserInterestedInCandidateKnownFor.HasIntersection

import com.twitter.ml.featurestore.catalog.features.onboarding.SimclusterUserInterestedInCandidateKnownFor.IntersectionCandidateKnownForScore

import com.twitter.ml.featurestore.catalog.features.onboarding.SimclusterUserInterestedInCandidateKnownFor.IntersectionClusterIds

import com.twitter.ml.featurestore.catalog.features.onboarding.SimclusterUserInterestedInCandidateKnownFor.IntersectionUserFavCandidateKnownForScore

import com.twitter.ml.featurestore.catalog.features.onboarding.SimclusterUserInterestedInCandidateKnownFor.IntersectionUserFavScore

import com.twitter.ml.featurestore.catalog.features.onboarding.SimclusterUserInterestedInCandidateKnownFor.IntersectionUserFollowCandidateKnownForScore

import com.twitter.ml.featurestore.catalog.features.onboarding.SimclusterUserInterestedInCandidateKnownFor.IntersectionUserFollowScore

import com.twitter.ml.featurestore.catalog.features.onboarding.UserWtfAlgorithmAggregate

import com.twitter.ml.featurestore.catalog.features.onboarding.WhoToFollowImpression.HomeTimelineWtfCandidateCounts

import com.twitter.ml.featurestore.catalog.features.onboarding.WhoToFollowImpression.HomeTimelineWtfCandidateImpressionCounts

import com.twitter.ml.featurestore.catalog.features.onboarding.WhoToFollowImpression.HomeTimelineWtfCandidateImpressionLatestTimestamp

import com.twitter.ml.featurestore.catalog.features.onboarding.WhoToFollowImpression.HomeTimelineWtfLatestTimestamp

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.FollowRate

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.Follows

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.FollowsTweetFavRate

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.FollowsTweetReplies

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.FollowsTweetReplyRate

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.FollowsTweetRetweetRate

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.FollowsTweetRetweets

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.FollowsWithTweetFavs

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.FollowsWithTweetImpressions

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.HasAnyEngagements

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.HasForwardEngagements

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.HasReverseEngagements

import com.twitter.ml.featurestore.catalog.features.onboarding.WtfUserAlgorithmAggregate.Impressions

import com.twitter.ml.featurestore.catalog.features.rux.UserResurrection.DaysSinceRecentResurrection

import com.twitter.ml.featurestore.catalog.features.timelines.AuthorTopicAggregates

import com.twitter.ml.featurestore.catalog.features.timelines.EngagementsReceivedByAuthorRealTimeAggregates

import com.twitter.ml.featurestore.catalog.features.timelines.NegativeEngagementsReceivedByAuthorRealTimeAggregates

import com.twitter.ml.featurestore.catalog.features.timelines.OriginalAuthorAggregates

import com.twitter.ml.featurestore.catalog.features.timelines.TopicEngagementRealTimeAggregates

import com.twitter.ml.featurestore.catalog.features.timelines.TopicEngagementUserStateRealTimeAggregates

import com.twitter.ml.featurestore.catalog.features.timelines.TopicNegativeEngagementUserStateRealTimeAggregates

import com.twitter.ml.featurestore.catalog.features.timelines.UserEngagementAuthorUserStateRealTimeAggregates

import com.twitter.ml.featurestore.catalog.features.timelines.UserNegativeEngagementAuthorUserStateRealTimeAggregates

import com.twitter.ml.featurestore.lib.EntityId

import com.twitter.ml.featurestore.lib.UserId

import com.twitter.ml.featurestore.lib.feature.BoundFeature

import com.twitter.ml.featurestore.lib.feature.Feature

object FeatureStoreFeatures {

import FeatureStoreRawFeatures.\_

///////////////////////////// Target user features ////////////////////////

val targetUserFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

(userKeyedFeatures ++ userAlgorithmAggregateFeatures).map(\_.bind(UserEntity))

val targetUserResurrectionFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

userResurrectionFeatures.map(\_.bind(UserEntity))

val targetUserWtfImpressionFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

wtfImpressionUserFeatures.map(\_.bind(UserEntity))

val targetUserUserAuthorUserStateRealTimeAggregatesFeature: Set[BoundFeature[\_ <: EntityId, \_]] =

userAuthorUserStateRealTimeAggregatesFeature.map(\_.bind(UserEntity))

val targetUserStatusFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

userStatusFeatures.map(\_.bind(UserEntity).logarithm1p)

val targetUserMetricCountFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

mcFeatures.map(\_.bind(UserEntity).logarithm1p)

val targetUserClientFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

clientFeatures.map(\_.bind(UserEntity))

///////////////////////////// Candidate user features ////////////////////////

val candidateUserFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

userKeyedFeatures.map(\_.bind(CandidateUserEntity))

val candidateUserAuthorRealTimeAggregateFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

authorAggregateFeatures.map(\_.bind(CandidateUserEntity))

val candidateUserResurrectionFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

userResurrectionFeatures.map(\_.bind(CandidateUserEntity))

val candidateUserStatusFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

userStatusFeatures.map(\_.bind(CandidateUserEntity).logarithm1p)

val candidateUserTimelinesAuthorAggregateFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

Set(timelinesAuthorAggregateFeatures.bind(CandidateUserEntity))

val candidateUserMetricCountFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

mcFeatures.map(\_.bind(CandidateUserEntity).logarithm1p)

val candidateUserClientFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

clientFeatures.map(\_.bind(CandidateUserEntity))

val similarToUserFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

(userKeyedFeatures ++ authorAggregateFeatures).map(\_.bind(AuthorEntity))

val similarToUserStatusFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

userStatusFeatures.map(\_.bind(AuthorEntity).logarithm1p)

val similarToUserTimelinesAuthorAggregateFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

Set(timelinesAuthorAggregateFeatures.bind(AuthorEntity))

val similarToUserMetricCountFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

mcFeatures.map(\_.bind(AuthorEntity).logarithm1p)

val userCandidateEdgeFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

(simclusterUVIntersectionFeatures ++ userCandidatePostNuxEdgeFeatures).map(

\_.bind(UserCandidateEntity))

val userCandidateWtfImpressionCandidateFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

wtfImpressionCandidateFeatures.map(\_.bind(UserCandidateEntity))

/\*\*

\* Aggregate features based on candidate source algorithms.

\*/

val postNuxAlgorithmIdAggregateFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

Set(PostNuxAlgorithmIdAggregateFeatureGroup.FeaturesAsDataRecord)

.map(\_.bind(WtfAlgorithmIdEntity))

/\*\*

\* Aggregate features based on candidate source algorithm types. There are 4 at the moment:

\* Geo, Social, Activity and Interest.

\*/

val postNuxAlgorithmTypeAggregateFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

Set(PostNuxAlgorithmTypeAggregateFeatureGroup.FeaturesAsDataRecord)

.map(\_.bind(WtfAlgorithmTypeEntity))

// user wtf-Algorithm features

val userWtfAlgorithmEdgeFeatures: Set[BoundFeature[\_ <: EntityId, \_]] =

FeatureGroupUtils.getTimelinesAggregationFrameworkCombinedFeatures(

UserWtfAlgorithmAggregate,

UserWtfAlgorithmEntity,

FeatureGroupUtils.getMaxSumAvgAggregate(UserWtfAlgorithmAggregate)

)

/\*\*

\* We have to add the max/sum/avg-aggregated features to the set of all features so that we can

\* register them using FRS's [[FrsFeatureJsonExporter]].

\*

\* Any additional such aggregated features that are included in [[FeatureStoreSource]] client

\* should be registered here as well.

\*/

val maxSumAvgAggregatedFeatureContext: FeatureContext = new FeatureContext()

.addFeatures(

UserWtfAlgorithmAggregate.getSecondaryAggregatedFeatureContext

)

// topic features

val topicAggregateFeatures: Set[BoundFeature[\_ <: EntityId, \_]] = Set(

TopicEngagementRealTimeAggregates.FeaturesAsDataRecord,

TopicNegativeEngagementUserStateRealTimeAggregates.FeaturesAsDataRecord,

TopicEngagementUserStateRealTimeAggregates.FeaturesAsDataRecord

).map(\_.bind(TopicEntity))

val userTopicFeatures: Set[BoundFeature[\_ <: EntityId, \_]] = Set(FollowedTopics.bind(UserEntity))

val authorTopicFeatures: Set[BoundFeature[\_ <: EntityId, \_]] = Set(

AuthorTopicAggregates.FeaturesAsDataRecord.bind(AuthorTopicEntity))

val topicFeatures = topicAggregateFeatures ++ userTopicFeatures ++ authorTopicFeatures

}

object FeatureStoreRawFeatures {

val mcFeatures = Set(

NumTweets,

NumRetweets,

NumOriginalTweets,

NumRetweetsReceived,

NumFavoritesReceived,

NumRepliesReceived,

NumQuoteRetweetsReceived,

NumFollowsReceived,

NumFollowBacks,

NumFollows,

NumUnfollows,

NumUnfollowBacks,

NumQualityFollowReceived,

NumQuoteRetweets,

NumFavorites,

NumReplies,

NumLoginTweetImpressions,

NumTweetImpressions,

NumLoginDays,

NumUserActiveMinutes,

NumMuted,

NumSpamBlocked,

NumMuteBacks,

NumSpamBlockedBacks,

NumWasMutualFollowed,

NumWasMutualUnfollowed,

NumWasUnfollowed

)

// based off usersource, and each feature represents the cumulative 'sent' counts

val userStatusFeatures = Set(

Favorites,

Followers,

Following,

Tweets

)

// ratio features created from combining other features

val userRatioFeatures = Set(MutualFollowsFollowing, FollowersFollowings)

// features related to user login history

val userResurrectionFeatures: Set[Feature[UserId, Int]] = Set(

DaysSinceRecentResurrection

)

// real-time aggregate features borrowed from timelines

val authorAggregateFeatures = Set(

EngagementsReceivedByAuthorRealTimeAggregates.FeaturesAsDataRecord,

NegativeEngagementsReceivedByAuthorRealTimeAggregates.FeaturesAsDataRecord,

)

val timelinesAuthorAggregateFeatures = OriginalAuthorAggregates.FeaturesAsDataRecord

val userAuthorUserStateRealTimeAggregatesFeature: Set[Feature[UserId, DataRecord]] = Set(

UserEngagementAuthorUserStateRealTimeAggregates.FeaturesAsDataRecord,

UserNegativeEngagementAuthorUserStateRealTimeAggregates.FeaturesAsDataRecord

)

// post nux per-user offline features

val userOfflineFeatures = Set(

NumFollowings,

NumFollowers,

NumMutualFollows,

TweepCred,

UserState,

Language,

Country,

MutualFollowsOverFollowingRatio,

MutualFollowsOverFollowersRatio,

FollowersOverFollowingRatio,

)

// matched post nux offline features between user and candidate

val userCandidatePostNuxEdgeFeatures = Set(

HaveSameUserState,

HaveSameLanguage,

HaveSameCountry,

NumFollowingsGap,

NumFollowersGap,

NumMutualFollowsGap,

TweepCredGap,

)

// user algorithm aggregate features

val userAlgorithmAggregateFeatures = Set(

Impressions,

Follows,

FollowRate,

FollowsWithTweetImpressions,

FollowsWithTweetFavs,

FollowsTweetFavRate,

FollowsTweetReplies,

FollowsTweetReplyRate,

FollowsTweetRetweets,

FollowsTweetRetweetRate,

HasForwardEngagements,

HasReverseEngagements,

HasAnyEngagements,

)

val userKeyedFeatures = userRatioFeatures ++ userOfflineFeatures

val wtfImpressionUserFeatures =

Set(HomeTimelineWtfCandidateCounts, HomeTimelineWtfLatestTimestamp)

val wtfImpressionCandidateFeatures =

Set(HomeTimelineWtfCandidateImpressionCounts, HomeTimelineWtfCandidateImpressionLatestTimestamp)

val simclusterUVIntersectionFeatures = Set(

IntersectionClusterIds,

HasIntersection,

IntersectionUserFollowScore,

IntersectionUserFavScore,

IntersectionCandidateKnownForScore,

IntersectionUserFollowCandidateKnownForScore,

IntersectionUserFavCandidateKnownForScore

)

// Client features

val clientFeatures = Set(

NumClients,

PrimaryClient,

PrimaryClientVersion,

FullPrimaryClientVersion,

PrimaryDeviceManufacturer,

PrimaryMobileSdkVersion,

SecondaryClient

)

}