package com.twitter.home\_mixer.product.for\_you

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

import com.twitter.home\_mixer.product.for\_you.candidate\_source.ScoredTweetWithConversationMetadata

import com.twitter.product\_mixer.core.feature.Feature

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.transformer.CandidateFeatureTransformer

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

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.BasicTopicContextFunctionalityType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.RecWithEducationTopicContextFunctionalityType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.RecommendationTopicContextFunctionalityType

import com.twitter.timelines.render.{thriftscala => tl}

import com.twitter.timelineservice.suggests.{thriftscala => tls}

object ForYouScoredTweetsResponseFeatureTransformer

extends CandidateFeatureTransformer[ScoredTweetWithConversationMetadata] {

override val identifier: TransformerIdentifier =

TransformerIdentifier("ForYouScoredTweetsResponse")

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

AncestorsFeature,

AuthorIdFeature,

AuthorIsBlueVerifiedFeature,

AuthorIsCreatorFeature,

AuthorIsGoldVerifiedFeature,

AuthorIsGrayVerifiedFeature,

AuthorIsLegacyVerifiedFeature,

ConversationModuleFocalTweetIdFeature,

ConversationModuleIdFeature,

DirectedAtUserIdFeature,

ExclusiveConversationAuthorIdFeature,

FullScoringSucceededFeature,

FavoritedByUserIdsFeature,

FollowedByUserIdsFeature,

InNetworkFeature,

InReplyToTweetIdFeature,

InReplyToUserIdFeature,

IsAncestorCandidateFeature,

IsReadFromCacheFeature,

IsRetweetFeature,

PerspectiveFilteredLikedByUserIdsFeature,

QuotedTweetIdFeature,

QuotedUserIdFeature,

SGSValidFollowedByUserIdsFeature,

SGSValidLikedByUserIdsFeature,

ScoreFeature,

SourceTweetIdFeature,

SourceUserIdFeature,

StreamToKafkaFeature,

SuggestTypeFeature,

TopicContextFunctionalityTypeFeature,

TopicIdSocialContextFeature

)

override def transform(input: ScoredTweetWithConversationMetadata): FeatureMap =

FeatureMapBuilder()

.add(AncestorsFeature, input.ancestors.getOrElse(Seq.empty))

.add(AuthorIdFeature, Some(input.authorId))

.add(AuthorIsBlueVerifiedFeature, input.authorIsBlueVerified.getOrElse(false))

.add(AuthorIsGoldVerifiedFeature, input.authorIsGoldVerified.getOrElse(false))

.add(AuthorIsGrayVerifiedFeature, input.authorIsGrayVerified.getOrElse(false))

.add(AuthorIsLegacyVerifiedFeature, input.authorIsLegacyVerified.getOrElse(false))

.add(AuthorIsCreatorFeature, input.authorIsCreator.getOrElse(false))

.add(ConversationModuleIdFeature, input.conversationId)

.add(ConversationModuleFocalTweetIdFeature, input.conversationFocalTweetId)

.add(DirectedAtUserIdFeature, input.directedAtUserId)

.add(ExclusiveConversationAuthorIdFeature, input.exclusiveConversationAuthorId)

.add(SGSValidLikedByUserIdsFeature, input.sgsValidLikedByUserIds.getOrElse(Seq.empty))

.add(SGSValidFollowedByUserIdsFeature, input.sgsValidFollowedByUserIds.getOrElse(Seq.empty))

.add(FavoritedByUserIdsFeature, input.sgsValidLikedByUserIds.getOrElse(Seq.empty))

.add(FollowedByUserIdsFeature, input.sgsValidFollowedByUserIds.getOrElse(Seq.empty))

.add(FullScoringSucceededFeature, true)

.add(InNetworkFeature, input.inNetwork.getOrElse(true))

.add(InReplyToTweetIdFeature, input.inReplyToTweetId)

.add(InReplyToUserIdFeature, input.inReplyToUserId)

.add(IsAncestorCandidateFeature, input.conversationFocalTweetId.exists(\_ != input.tweetId))

.add(IsReadFromCacheFeature, input.isReadFromCache.getOrElse(false))

.add(IsRetweetFeature, input.sourceTweetId.isDefined)

.add(

PerspectiveFilteredLikedByUserIdsFeature,

input.perspectiveFilteredLikedByUserIds.getOrElse(Seq.empty))

.add(QuotedTweetIdFeature, input.quotedTweetId)

.add(QuotedUserIdFeature, input.quotedUserId)

.add(ScoreFeature, input.score)

.add(SourceTweetIdFeature, input.sourceTweetId)

.add(SourceUserIdFeature, input.sourceUserId)

.add(StreamToKafkaFeature, input.streamToKafka.getOrElse(false))

.add(SuggestTypeFeature, input.suggestType.orElse(Some(tls.SuggestType.Undefined)))

.add(

TopicContextFunctionalityTypeFeature,

input.topicFunctionalityType.collect {

case tl.TopicContextFunctionalityType.Basic => BasicTopicContextFunctionalityType

case tl.TopicContextFunctionalityType.Recommendation =>

RecommendationTopicContextFunctionalityType

case tl.TopicContextFunctionalityType.RecWithEducation =>

RecWithEducationTopicContextFunctionalityType

}

)

.add(TopicIdSocialContextFeature, input.topicId)

.build()

}