package com.twitter.home\_mixer.candidate\_pipeline

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

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

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

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

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

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.search.earlybird.{thriftscala => t}

object FollowingEarlybirdResponseFeatureTransformer

extends CandidateFeatureTransformer[t.ThriftSearchResult] {

override val identifier: TransformerIdentifier =

TransformerIdentifier("FollowingEarlybirdResponse")

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

AuthorIdFeature,

InReplyToTweetIdFeature,

IsRetweetFeature,

SourceTweetIdFeature,

SourceUserIdFeature,

)

override def transform(candidate: t.ThriftSearchResult): FeatureMap = FeatureMapBuilder()

.add(AuthorIdFeature, candidate.tweetypieTweet.flatMap(\_.coreData.map(\_.userId)))

.add(

InReplyToTweetIdFeature,

candidate.tweetypieTweet.flatMap(\_.coreData.flatMap(\_.reply.flatMap(\_.inReplyToStatusId))))

.add(IsRetweetFeature, candidate.metadata.exists(\_.isRetweet.contains(true)))

.add(SourceTweetIdFeature, candidate.sourceTweetypieTweet.map(\_.id))

.add(SourceUserIdFeature, candidate.sourceTweetypieTweet.flatMap(\_.coreData.map(\_.userId)))

.build()

}