package com.twitter.timelineranker.common

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.servo.util.FutureArrow

import com.twitter.timelineranker.core.HydratedCandidatesAndFeaturesEnvelope

import com.twitter.timelineranker.model.CandidateTweet

import com.twitter.timelineranker.model.CandidateTweetsResult

import com.twitter.util.Future

class CandidateGenerationTransform(statsReceiver: StatsReceiver)

extends FutureArrow[HydratedCandidatesAndFeaturesEnvelope, CandidateTweetsResult] {

private[this] val numCandidateTweetsStat = statsReceiver.stat("numCandidateTweets")

private[this] val numSourceTweetsStat = statsReceiver.stat("numSourceTweets")

override def apply(

candidatesAndFeaturesEnvelope: HydratedCandidatesAndFeaturesEnvelope

): Future[CandidateTweetsResult] = {

val hydratedTweets = candidatesAndFeaturesEnvelope.candidateEnvelope.hydratedTweets.outerTweets

if (hydratedTweets.nonEmpty) {

val candidates = hydratedTweets.map { hydratedTweet =>

CandidateTweet(hydratedTweet, candidatesAndFeaturesEnvelope.features(hydratedTweet.tweetId))

}

numCandidateTweetsStat.add(candidates.size)

val sourceTweets =

candidatesAndFeaturesEnvelope.candidateEnvelope.sourceHydratedTweets.outerTweets.map {

hydratedTweet =>

CandidateTweet(

hydratedTweet,

candidatesAndFeaturesEnvelope.features(hydratedTweet.tweetId))

}

numSourceTweetsStat.add(sourceTweets.size)

Future.value(CandidateTweetsResult(candidates, sourceTweets))

} else {

Future.value(CandidateTweetsResult.Empty)

}

}

}