package com.twitter.frigate.pushservice.rank

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.frigate.common.base.CandidateDetails

import com.twitter.frigate.pushservice.model.PushTypes.PushCandidate

import com.twitter.frigate.thriftscala.CommonRecommendationType

/\*\*

\* This Ranker re-ranks MR candidates, down ranks input CRTs.

\* Relative ranking between input CRTs and rest of the candidates doesn't change

\*

\* Ex: T: Tweet candidate, F: input CRT candidates

\*

\* T3, F2, T1, T2, F1 => T3, T1, T2, F2, F1

\*/

case class CRTDownRanker(statsReceiver: StatsReceiver) {

private val recsToDownRankStat = statsReceiver.stat("recs\_to\_down\_rank")

private val otherRecsStat = statsReceiver.stat("other\_recs")

private val downRankerRequests = statsReceiver.counter("down\_ranker\_requests")

private def downRank(

inputCandidates: Seq[CandidateDetails[PushCandidate]],

crtToDownRank: CommonRecommendationType

): Seq[CandidateDetails[PushCandidate]] = {

downRankerRequests.incr()

val (downRankedCandidates, otherCandidates) =

inputCandidates.partition(\_.candidate.commonRecType == crtToDownRank)

recsToDownRankStat.add(downRankedCandidates.size)

otherRecsStat.add(otherCandidates.size)

otherCandidates ++ downRankedCandidates

}

final def downRank(

inputCandidates: Seq[CandidateDetails[PushCandidate]],

crtsToDownRank: Seq[CommonRecommendationType]

): Seq[CandidateDetails[PushCandidate]] = {

crtsToDownRank.headOption match {

case Some(crt) =>

val downRankedCandidates = downRank(inputCandidates, crt)

downRank(downRankedCandidates, crtsToDownRank.tail)

case None => inputCandidates

}

}

}