package com.twitter.visibility.builder.users

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.visibility.builder.FeatureMapBuilder

import com.twitter.visibility.features.AuthorBlocksOuterAuthor

import com.twitter.visibility.features.OuterAuthorFollowsAuthor

import com.twitter.visibility.features.OuterAuthorId

import com.twitter.visibility.features.OuterAuthorIsInnerAuthor

class QuotedTweetFeatures(

relationshipFeatures: RelationshipFeatures,

statsReceiver: StatsReceiver) {

private[this] val scopedStatsReceiver = statsReceiver.scope("quoted\_tweet\_features")

private[this] val requests = scopedStatsReceiver.counter("requests")

private[this] val outerAuthorIdStat =

scopedStatsReceiver.scope(OuterAuthorId.name).counter("requests")

private[this] val authorBlocksOuterAuthor =

scopedStatsReceiver.scope(AuthorBlocksOuterAuthor.name).counter("requests")

private[this] val outerAuthorFollowsAuthor =

scopedStatsReceiver.scope(OuterAuthorFollowsAuthor.name).counter("requests")

private[this] val outerAuthorIsInnerAuthor =

scopedStatsReceiver.scope(OuterAuthorIsInnerAuthor.name).counter("requests")

def forOuterAuthor(

outerAuthorId: Long,

innerAuthorId: Long

): FeatureMapBuilder => FeatureMapBuilder = {

requests.incr()

outerAuthorIdStat.incr()

authorBlocksOuterAuthor.incr()

outerAuthorFollowsAuthor.incr()

outerAuthorIsInnerAuthor.incr()

val viewer = Some(outerAuthorId)

\_.withConstantFeature(OuterAuthorId, outerAuthorId)

.withFeature(

AuthorBlocksOuterAuthor,

relationshipFeatures.authorBlocksViewer(innerAuthorId, viewer))

.withFeature(

OuterAuthorFollowsAuthor,

relationshipFeatures.viewerFollowsAuthor(innerAuthorId, viewer))

.withConstantFeature(

OuterAuthorIsInnerAuthor,

innerAuthorId == outerAuthorId

)

}

}