package com.twitter.follow\_recommendations.common.models

import com.twitter.simclusters\_v2.thriftscala.InternalId

import com.twitter.usersignalservice.thriftscala.SignalType

import com.twitter.usersignalservice.thriftscala.Signal

trait SignalData {

val userId: Long

val signalType: SignalType

}

case class RecentFollowsSignal(

override val userId: Long,

override val signalType: SignalType,

followedUserId: Long,

timestamp: Long)

extends SignalData

object RecentFollowsSignal {

def fromUssSignal(targetUserId: Long, signal: Signal): RecentFollowsSignal = {

val InternalId.UserId(followedUserId) = signal.targetInternalId.getOrElse(

throw new IllegalArgumentException("RecentFollow Signal does not have internalId"))

RecentFollowsSignal(

userId = targetUserId,

followedUserId = followedUserId,

timestamp = signal.timestamp,

signalType = signal.signalType

)

}

def getRecentFollowedUserIds(

signalDataMap: Option[Map[SignalType, Seq[SignalData]]]

): Option[Seq[Long]] = {

signalDataMap.map(\_.getOrElse(SignalType.AccountFollow, default = Seq.empty).flatMap {

case RecentFollowsSignal(userId, signalType, followedUserId, timestamp) =>

Some(followedUserId)

case \_ => None

})

}

}