package com.twitter.product\_mixer.component\_library.feature\_hydrator.candidate.tweet\_tlx

import com.twitter.ml.featurestore.timelines.thriftscala.TimelineScorerScoreView

import com.twitter.product\_mixer.component\_library.model.candidate.TweetCandidate

import com.twitter.product\_mixer.component\_library.scorer.tweet\_tlx.TLXScore

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.feature\_hydrator.CandidateFeatureHydrator

import com.twitter.product\_mixer.core.model.common.identifier.FeatureHydratorIdentifier

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.stitch.Stitch

import com.twitter.strato.generated.client.ml.featureStore.TimelineScorerTweetScoresV1ClientColumn

import com.twitter.timelinescorer.thriftscala.v1

import javax.inject.Inject

import javax.inject.Singleton

/\*\*

\* Hydrate Tweet Scores via Timeline Scorer (TLX)

\*

\* Note that this is the [[CandidateFeatureHydrator]] version of

\* [[com.twitter.product\_mixer.component\_library.scorer.tweet\_tlx.TweetTLXStratoScorer]]

\*/

@Singleton

class TweetTLXScoreCandidateFeatureHydrator @Inject() (

column: TimelineScorerTweetScoresV1ClientColumn)

extends CandidateFeatureHydrator[PipelineQuery, TweetCandidate] {

override val identifier: FeatureHydratorIdentifier =

FeatureHydratorIdentifier("TweetTLXScore")

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

private val NoScoreMap = FeatureMapBuilder()

.add(TLXScore, None)

.build()

override def apply(

query: PipelineQuery,

candidate: TweetCandidate,

existingFeatures: FeatureMap

): Stitch[FeatureMap] = {

query.getOptionalUserId match {

case Some(userId) =>

column.fetcher

.fetch(candidate.id, TimelineScorerScoreView(Some(userId)))

.map(scoredTweet =>

scoredTweet.v match {

case Some(v1.ScoredTweet(Some(\_), score, \_, \_)) =>

FeatureMapBuilder()

.add(TLXScore, score)

.build()

case \_ => throw new Exception(s"Invalid response from TLX: ${scoredTweet.v}")

})

case \_ =>

Stitch.value(NoScoreMap)

}

}

}