package com.twitter.frigate.pushservice.model

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.frigate.common.base.TopicProofTweetCandidate

import com.twitter.frigate.common.base.TweetAuthorDetails

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

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

import com.twitter.frigate.pushservice.config.Config

import com.twitter.frigate.pushservice.ml.PushMLModelScorer

import com.twitter.frigate.pushservice.model.candidate.CopyIds

import com.twitter.frigate.pushservice.model.ibis.TopicProofTweetIbis2Hydrator

import com.twitter.frigate.pushservice.model.ntab.TopicProofTweetNtabRequestHydrator

import com.twitter.frigate.pushservice.params.PushFeatureSwitchParams

import com.twitter.frigate.pushservice.predicate.PredicatesForCandidate

import com.twitter.frigate.pushservice.take.predicates.BasicTweetPredicatesForRFPH

import com.twitter.frigate.thriftscala.CommonRecommendationType

import com.twitter.gizmoduck.thriftscala.User

import com.twitter.hermit.predicate.NamedPredicate

import com.twitter.stitch.tweetypie.TweetyPie

import com.twitter.util.Future

/\*\*

\* This class defines a hydrated [[TopicProofTweetCandidate]]

\*

\* @param candidate : [[TopicProofTweetCandidate]] for the candidate representint a Tweet recommendation for followed Topic

\* @param author : Tweet author representated as Gizmoduck user object

\* @param copyIds : push and ntab notification copy

\* @param stats : finagle scoped states receiver

\* @param pushModelScorer : ML model score object for fetching prediction scores

\*/

class TopicProofTweetPushCandidate(

candidate: RawCandidate with TopicProofTweetCandidate,

author: Option[User],

copyIds: CopyIds

)(

implicit stats: StatsReceiver,

pushModelScorer: PushMLModelScorer)

extends PushCandidate

with TopicProofTweetCandidate

with TweetAuthorDetails

with TopicProofTweetNtabRequestHydrator

with TopicProofTweetIbis2Hydrator {

override val statsReceiver: StatsReceiver = stats

override val target: PushTypes.Target = candidate.target

override val tweetId: Long = candidate.tweetId

override lazy val tweetyPieResult: Option[TweetyPie.TweetyPieResult] = candidate.tweetyPieResult

override val weightedOpenOrNtabClickModelScorer: PushMLModelScorer = pushModelScorer

override val pushCopyId: Option[Int] = copyIds.pushCopyId

override val ntabCopyId: Option[Int] = copyIds.ntabCopyId

override val copyAggregationId: Option[String] = copyIds.aggregationId

override val semanticCoreEntityId = candidate.semanticCoreEntityId

override val localizedUttEntity = candidate.localizedUttEntity

override val tweetAuthor = Future.value(author)

override val topicListingSetting = candidate.topicListingSetting

override val algorithmCR = candidate.algorithmCR

override val commonRecType: CommonRecommendationType = candidate.commonRecType

override val tagsCR = candidate.tagsCR

override val isOutOfNetwork = candidate.isOutOfNetwork

}

case class TopicProofTweetCandidatePredicates(override val config: Config)

extends BasicTweetPredicatesForRFPH[TopicProofTweetPushCandidate] {

implicit val statsReceiver: StatsReceiver = config.statsReceiver.scope(getClass.getSimpleName)

override val preCandidateSpecificPredicates: List[NamedPredicate[TopicProofTweetPushCandidate]] =

List(

PredicatesForCandidate.paramPredicate(

PushFeatureSwitchParams.EnableTopicProofTweetRecs

),

)

}