package com.twitter.product\_mixer.component\_library.feature\_hydrator.query.cr\_ml\_ranker

import com.twitter.cr\_ml\_ranker.{thriftscala => t}

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.QueryFeatureHydrator

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

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

import com.twitter.stitch.Stitch

object CrMlRankerCommonFeatures extends Feature[PipelineQuery, t.CommonFeatures]

object CrMlRankerRankingConfig extends Feature[PipelineQuery, t.RankingConfig]

private[cr\_ml\_ranker] class CrMlRankerCommonQueryFeatureHydrator(

crMlRanker: t.CrMLRanker.MethodPerEndpoint,

rankingConfigSelector: RankingConfigBuilder)

extends QueryFeatureHydrator[PipelineQuery] {

override val identifier: FeatureHydratorIdentifier = FeatureHydratorIdentifier("CrMlRanker")

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

Set(CrMlRankerCommonFeatures, CrMlRankerRankingConfig)

override def hydrate(query: PipelineQuery): Stitch[FeatureMap] = {

val rankingConfig = rankingConfigSelector.apply(query)

Stitch

.callFuture(

crMlRanker.getCommonFeatures(

t.RankingRequestContext(query.getRequiredUserId, rankingConfig))).map { commonFeatures =>

FeatureMapBuilder()

.add(CrMlRankerRankingConfig, rankingConfig)

.add(CrMlRankerCommonFeatures, commonFeatures)

.build()

}

}

}