package com.twitter.home\_mixer.product.scored\_tweets.scorer

import com.twitter.dal.personal\_data.{thriftjava => pd}

import com.twitter.home\_mixer.product.scored\_tweets.param.ScoredTweetsParam.Scoring.ModelWeights

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

import com.twitter.product\_mixer.core.feature.datarecord.DataRecordOptionalFeature

import com.twitter.product\_mixer.core.feature.datarecord.DoubleDataRecordCompatible

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

import com.twitter.timelines.configapi.FSBoundedParam

import com.twitter.timelines.prediction.features.recap.RecapFeatures

sealed trait PredictedScoreFeature

extends DataRecordOptionalFeature[TweetCandidate, Double]

with DoubleDataRecordCompatible {

override val personalDataTypes: Set[pd.PersonalDataType] = Set.empty

def statName: String

def modelWeightParam: FSBoundedParam[Double]

def extractScore: FeatureMap => Option[Double] = \_.getOrElse(this, None)

}

object PredictedFavoriteScoreFeature extends PredictedScoreFeature {

override val featureName: String = RecapFeatures.PREDICTED\_IS\_FAVORITED.getFeatureName

override val statName = "fav"

override val modelWeightParam = ModelWeights.FavParam

}

object PredictedReplyScoreFeature extends PredictedScoreFeature {

override val featureName: String = RecapFeatures.PREDICTED\_IS\_REPLIED.getFeatureName

override val statName = "reply"

override val modelWeightParam = ModelWeights.ReplyParam

}

object PredictedRetweetScoreFeature extends PredictedScoreFeature {

override val featureName: String = RecapFeatures.PREDICTED\_IS\_RETWEETED.getFeatureName

override val statName = "retweet"

override val modelWeightParam = ModelWeights.RetweetParam

}

object PredictedReplyEngagedByAuthorScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_REPLIED\_REPLY\_ENGAGED\_BY\_AUTHOR.getFeatureName

override val statName = "reply\_engaged\_by\_author"

override val modelWeightParam = ModelWeights.ReplyEngagedByAuthorParam

}

object PredictedGoodClickConvoDescFavoritedOrRepliedScoreFeature extends PredictedScoreFeature {

override val featureName: String = RecapFeatures.PREDICTED\_IS\_GOOD\_CLICKED\_V1.getFeatureName

override val statName = "good\_click\_convo\_desc\_favorited\_or\_replied"

override val modelWeightParam = ModelWeights.GoodClickParam

override def extractScore: FeatureMap => Option[Double] = { featureMap =>

val goodClickV1Opt = featureMap.getOrElse(this, None)

val goodClickV2Opt = featureMap.getOrElse(PredictedGoodClickConvoDescUamGt2ScoreFeature, None)

(goodClickV1Opt, goodClickV2Opt) match {

case (Some(v1Score), Some(v2Score)) => Some(Math.max(v1Score, v2Score))

case \_ => goodClickV1Opt.orElse(goodClickV2Opt)

}

}

}

object PredictedGoodClickConvoDescUamGt2ScoreFeature extends PredictedScoreFeature {

override val featureName: String = RecapFeatures.PREDICTED\_IS\_GOOD\_CLICKED\_V2.getFeatureName

override val statName = "good\_click\_convo\_desc\_uam\_gt\_2"

override val modelWeightParam = ModelWeights.GoodClickV2Param

}

object PredictedGoodProfileClickScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_PROFILE\_CLICKED\_AND\_PROFILE\_ENGAGED.getFeatureName

override val statName = "good\_profile\_click"

override val modelWeightParam = ModelWeights.GoodProfileClickParam

}

object PredictedVideoPlayback50ScoreFeature extends PredictedScoreFeature {

override val featureName: String = RecapFeatures.PREDICTED\_IS\_VIDEO\_PLAYBACK\_50.getFeatureName

override val statName = "video\_playback\_50"

override val modelWeightParam = ModelWeights.VideoPlayback50Param

}

object PredictedTweetDetailDwellScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_TWEET\_DETAIL\_DWELLED\_15\_SEC.getFeatureName

override val statName = "tweet\_detail\_dwell"

override val modelWeightParam = ModelWeights.TweetDetailDwellParam

}

object PredictedProfileDwelledScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_PROFILE\_DWELLED\_20\_SEC.getFeatureName

override val statName = "profile\_dwell"

override val modelWeightParam = ModelWeights.ProfileDwelledParam

}

object PredictedBookmarkScoreFeature extends PredictedScoreFeature {

override val featureName: String = RecapFeatures.PREDICTED\_IS\_BOOKMARKED.getFeatureName

override val statName = "bookmark"

override val modelWeightParam = ModelWeights.BookmarkParam

}

object PredictedShareScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_SHARED.getFeatureName

override val statName = "share"

override val modelWeightParam = ModelWeights.ShareParam

}

object PredictedShareMenuClickScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_SHARE\_MENU\_CLICKED.getFeatureName

override val statName = "share\_menu\_click"

override val modelWeightParam = ModelWeights.ShareMenuClickParam

}

// Negative Engagements

object PredictedNegativeFeedbackV2ScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_NEGATIVE\_FEEDBACK\_V2.getFeatureName

override val statName = "negative\_feedback\_v2"

override val modelWeightParam = ModelWeights.NegativeFeedbackV2Param

}

object PredictedReportedScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_REPORT\_TWEET\_CLICKED.getFeatureName

override val statName = "reported"

override val modelWeightParam = ModelWeights.ReportParam

}

object PredictedStrongNegativeFeedbackScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_STRONG\_NEGATIVE\_FEEDBACK.getFeatureName

override val statName = "strong\_negative\_feedback"

override val modelWeightParam = ModelWeights.StrongNegativeFeedbackParam

}

object PredictedWeakNegativeFeedbackScoreFeature extends PredictedScoreFeature {

override val featureName: String =

RecapFeatures.PREDICTED\_IS\_WEAK\_NEGATIVE\_FEEDBACK.getFeatureName

override val statName = "weak\_negative\_feedback"

override val modelWeightParam = ModelWeights.WeakNegativeFeedbackParam

}

object PredictedScoreFeature {

val PredictedScoreFeatures: Set[PredictedScoreFeature] = Set(

PredictedFavoriteScoreFeature,

PredictedReplyScoreFeature,

PredictedRetweetScoreFeature,

PredictedReplyEngagedByAuthorScoreFeature,

PredictedGoodClickConvoDescFavoritedOrRepliedScoreFeature,

PredictedGoodClickConvoDescUamGt2ScoreFeature,

PredictedGoodProfileClickScoreFeature,

PredictedVideoPlayback50ScoreFeature,

PredictedTweetDetailDwellScoreFeature,

PredictedProfileDwelledScoreFeature,

PredictedBookmarkScoreFeature,

PredictedShareScoreFeature,

PredictedShareMenuClickScoreFeature,

// Negative Engagements

PredictedNegativeFeedbackV2ScoreFeature,

PredictedReportedScoreFeature,

PredictedStrongNegativeFeedbackScoreFeature,

PredictedWeakNegativeFeedbackScoreFeature,

)

}