package com.twitter.timelines.prediction.features.followsource

import com.twitter.ml.api.Feature

import com.twitter.dal.personal\_data.thriftjava.PersonalDataType.\_

import scala.collection.JavaConverters.\_

object FollowSourceFeatures {

// Corresponds to an algorithm constant from com.twitter.hermit.profile.HermitProfileConstants

val FollowSourceAlgorithm = new Feature.Text("follow\_source.algorithm")

// Type of follow action: one of "unfollow", "follow", "follow\_back", "follow\_many", "follow\_all"

val FollowAction = new Feature.Text(

"follow\_source.action",

Set(Follow, PrivateAccountsFollowedBy, PublicAccountsFollowedBy).asJava)

// Millisecond timestamp when follow occurred

val FollowTimestamp =

new Feature.Discrete("follow\_source.follow\_timestamp", Set(Follow, PrivateTimestamp).asJava)

// Age of follow (in minutes)

val FollowAgeMinutes =

new Feature.Continuous("follow\_source.follow\_age\_minutes", Set(Follow).asJava)

// Tweet ID of tweet details page from where follow happened (if applicable)

val FollowCauseTweetId = new Feature.Discrete("follow\_source.cause\_tweet\_id", Set(TweetId).asJava)

// String representation of follow client (android, web, iphone, etc). Derived from "client"

// portion of client event namespace.

val FollowClientId = new Feature.Text("follow\_source.client\_id", Set(ClientType).asJava)

// If the follow happens via a profile's Following or Followers,

// the id of the profile owner is recorded here.

val FollowAssociationId =

new Feature.Discrete("follow\_source.association\_id", Set(Follow, UserId).asJava)

// The "friendly name" here is computed using FollowSourceUtil.getSource. It represents

// a grouping on a few client events that reflect where the event occurred. For example,

// events on the tweet details page are grouped using "tweetDetails":

// case (Some("web"), Some("permalink"), \_, \_, \_) => "tweetDetails"

// case (Some("iphone"), Some("tweet"), \_, \_, \_) => "tweetDetails"

// case (Some("android"), Some("tweet"), \_, \_, \_) => "tweetDetails"

val FollowSourceFriendlyName = new Feature.Text("follow\_source.friendly\_name", Set(Follow).asJava)

// Up to two sources and actions that preceded the follow (for example, a profile visit

// through a mention click, which itself was on a tweet detail page reached through a tweet

// click in the Home tab). See go/followsource for more details and examples.

// The "source" here is computed using FollowSourceUtil.getSource

val PreFollowAction1 = new Feature.Text("follow\_source.pre\_follow\_action\_1", Set(Follow).asJava)

val PreFollowAction2 = new Feature.Text("follow\_source.pre\_follow\_action\_2", Set(Follow).asJava)

val PreFollowSource1 = new Feature.Text("follow\_source.pre\_follow\_source\_1", Set(Follow).asJava)

val PreFollowSource2 = new Feature.Text("follow\_source.pre\_follow\_source\_2", Set(Follow).asJava)

}