package com.twitter.product\_mixer.core.feature.featurestorev1.featurevalue

import com.fasterxml.jackson.annotation.JsonIgnoreProperties

import com.fasterxml.jackson.annotation.JsonProperty

import com.twitter.ml.api.DataRecordMerger

import com.twitter.ml.api.util.SRichDataRecord

import com.twitter.ml.featurestore.lib.data.HydrationError

import com.twitter.product\_mixer.core.feature.Feature

private[product\_mixer] object FeatureStoreV1ResponseFeature

extends Feature[Any, FeatureStoreV1Response]

@JsonIgnoreProperties(Array("richDataRecord", "failedFeatures"))

private[product\_mixer] case class FeatureStoreV1Response(

@JsonProperty("richDataRecord") richDataRecord: SRichDataRecord,

@JsonProperty("failedFeatures") failedFeatures: Map[\_ <: Feature[\_, \_], Set[HydrationError]]) {

// Since RichDataRecord is Java, we need to override this.

override def equals(obj: Any): Boolean = obj match {

case that: FeatureStoreV1Response =>

failedFeatures == that.failedFeatures && richDataRecord.getRecord.equals(

that.richDataRecord.getRecord)

case \_ => false

}

}

private[product\_mixer] object FeatureStoreV1Response {

val dataRecordMerger = new DataRecordMerger

def merge(

left: FeatureStoreV1Response,

right: FeatureStoreV1Response

): FeatureStoreV1Response = {

val newDataRecord = left.richDataRecord.getRecord.deepCopy()

dataRecordMerger.merge(newDataRecord, right.richDataRecord.getRecord)

FeatureStoreV1Response(

richDataRecord = SRichDataRecord(newDataRecord),

left.failedFeatures ++ right.failedFeatures

)

}

}