package com.twitter.simclusters\_v2.common.ml

import com.twitter.ml.api.Feature.Continuous

import com.twitter.ml.api.Feature.SparseContinuous

import com.twitter.ml.api.\_

import com.twitter.ml.api.util.FDsl.\_

import com.twitter.simclusters\_v2.common.SimClustersEmbedding

class SimClustersEmbeddingAdapter(embeddingFeature: SparseContinuous)

extends IRecordOneToOneAdapter[SimClustersEmbedding] {

override def getFeatureContext: FeatureContext = new FeatureContext(embeddingFeature)

override def adaptToDataRecord(embedding: SimClustersEmbedding): DataRecord = {

val embeddingMap = embedding.embedding.map {

case (clusterId, score) =>

(clusterId.toString, score)

}.toMap

new DataRecord().setFeatureValue(embeddingFeature, embeddingMap)

}

}

class NormalizedSimClustersEmbeddingAdapter(

embeddingFeature: SparseContinuous,

normFeature: Continuous)

extends IRecordOneToOneAdapter[SimClustersEmbedding] {

override def getFeatureContext: FeatureContext = new FeatureContext(embeddingFeature, normFeature)

override def adaptToDataRecord(embedding: SimClustersEmbedding): DataRecord = {

val normalizedEmbedding = Map(

embedding.sortedClusterIds.map(\_.toString).zip(embedding.normalizedSortedScores): \_\*)

val dataRecord = new DataRecord().setFeatureValue(embeddingFeature, normalizedEmbedding)

dataRecord.setFeatureValue(normFeature, embedding.l2norm)

}

}