package com.twitter.product\_mixer.core.feature.datarecord

import com.twitter.ml.api.DataRecord

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

/\*\*

\* A DataRecord supported feature mixin for enabling conversions from Product Mixer Features

\* to DataRecords. When using Feature Store features, this is pre-configured for the customer

\* under the hood. For non-Feature Store features, customers must mix in either [[DataRecordFeature]]

\* for required features, or [[DataRecordOptionalFeature]] for optional features, as well as mixing

\* in a corresponding [[DataRecordCompatible]] for their feature type.

\* @tparam Entity The type of entity that this feature works with. This could be a User, Tweet,

\* Query, etc.

\* @tparam Value The type of the value of this feature.

\*/

sealed trait BaseDataRecordFeature[-Entity, Value] extends Feature[Entity, Value]

private[product\_mixer] abstract class FeatureStoreDataRecordFeature[-Entity, Value]

extends BaseDataRecordFeature[Entity, Value]

/\*\*

\* Feature in a DataRecord for a required feature value; the corresponding feature will always be

\* available in the built DataRecord.

\*/

trait DataRecordFeature[-Entity, Value] extends BaseDataRecordFeature[Entity, Value] {

self: DataRecordCompatible[Value] =>

}

/\*\*

\* Feature in a DataRecord for an optional feature value; the corresponding feature will only

\* ever be set in a DataRecord if the value in the feature map is defined (Some(V)).

\*/

trait DataRecordOptionalFeature[-Entity, Value]

extends BaseDataRecordFeature[Entity, Option[Value]] {

self: DataRecordCompatible[Value] =>

}

/\*\*

\* An entire DataRecord as a feature. This is useful when there is an existing DataRecord that

\* should be used as a whole instead of as individual [[DataRecordFeature]]s for example.

\*/

trait DataRecordInAFeature[-Entity] extends BaseDataRecordFeature[Entity, DataRecord]