#include "internal/thrift.h"

#include "internal/error.h"

#include <twml/DataRecordReader.h>

#include <twml/HashedDataRecordReader.h>

#include <twml/BatchPredictionRequest.h>

#include <twml/Error.h>

#include <algorithm>

#include <cstring>

#include <cstdint>

namespace twml {

template<typename RecordType>

void GenericBatchPredictionRequest<RecordType>::decode(Reader &reader) {

uint8\_t feature\_type = reader.readByte();

while (feature\_type != TTYPE\_STOP) {

int16\_t field\_id = reader.readInt16();

switch (field\_id) {

case 1: {

CHECK\_THRIFT\_TYPE(feature\_type, TTYPE\_LIST, "list");

CHECK\_THRIFT\_TYPE(reader.readByte(), TTYPE\_STRUCT, "list\_element");

int32\_t length = reader.readInt32();

m\_requests.resize(length, RecordType(this->num\_labels, this->num\_weights));

for (auto &request : m\_requests) {

request.decode(reader);

}

break;

}

case 2: {

CHECK\_THRIFT\_TYPE(feature\_type, TTYPE\_STRUCT, "commonFeatures");

m\_common\_features.decode(reader);

break;

}

default: throw ThriftInvalidField(field\_id, \_\_func\_\_);

}

feature\_type = reader.readByte();

}

return;

}

// Instantiate decoders.

template void GenericBatchPredictionRequest<HashedDataRecord>::decode(HashedDataRecordReader &reader);

template void GenericBatchPredictionRequest<DataRecord>::decode(DataRecordReader &reader);

} // namespace twml