package com.twitter.product\_mixer.component\_library.scorer.common

import com.twitter.finagle.Http

import com.twitter.finagle.grpc.FinagleChannelBuilder

import com.twitter.finagle.grpc.FutureConverters

import com.twitter.stitch.Stitch

import inference.GRPCInferenceServiceGrpc

import inference.GrpcService.ModelInferRequest

import inference.GrpcService.ModelInferResponse

import io.grpc.ManagedChannel

/\*\*

\* Client wrapper for calling a Cortex Managed Inference Service (go/cmis) ML Model using GRPC.

\* @param httpClient Finagle HTTP Client to use for connection.

\* @param modelPath Wily path to the ML Model service (e.g. /cluster/local/role/service/instance).

\*/

case class ManagedModelClient(

httpClient: Http.Client,

modelPath: String)

extends MLModelInferenceClient {

private val channel: ManagedChannel =

FinagleChannelBuilder.forTarget(modelPath).httpClient(httpClient).build()

private val inferenceServiceStub = GRPCInferenceServiceGrpc.newFutureStub(channel)

def score(request: ModelInferRequest): Stitch[ModelInferResponse] = {

Stitch

.callFuture(

FutureConverters

.RichListenableFuture(inferenceServiceStub.modelInfer(request)).toTwitter)

}

}