package com.twitter.product\_mixer.core.pipeline.scoring

import com.twitter.product\_mixer.core.functional\_component.scorer.ScoredCandidateResult

import com.twitter.product\_mixer.core.model.common.UniversalNoun

import com.twitter.product\_mixer.core.pipeline.PipelineResult

import com.twitter.product\_mixer.core.pipeline.pipeline\_failure.PipelineFailure

import com.twitter.product\_mixer.core.service.candidate\_feature\_hydrator\_executor.CandidateFeatureHydratorExecutorResult

import com.twitter.product\_mixer.core.service.gate\_executor.GateExecutorResult

import com.twitter.product\_mixer.core.service.selector\_executor.SelectorExecutorResult

/\*\*

\* The Results of every step during the ScoringPipeline process. The end result contains

\* only the candidates that were actually scored (e.g, not dropped by a filter) with an updated,

\* combined feature map of all features that were passed in with the candidate plus all features

\* returned as part of scoring.

\*/

case class ScoringPipelineResult[Candidate <: UniversalNoun[Any]](

gateResults: Option[GateExecutorResult],

selectorResults: Option[SelectorExecutorResult],

preScoringHydrationPhase1Result: Option[CandidateFeatureHydratorExecutorResult[Candidate]],

preScoringHydrationPhase2Result: Option[CandidateFeatureHydratorExecutorResult[Candidate]],

scorerResults: Option[CandidateFeatureHydratorExecutorResult[

Candidate

]],

failure: Option[PipelineFailure],

result: Option[Seq[ScoredCandidateResult[Candidate]]])

extends PipelineResult[Seq[ScoredCandidateResult[Candidate]]] {

override val resultSize: Int = result.map(\_.size).getOrElse(0)

override def withFailure(

failure: PipelineFailure

): ScoringPipelineResult[Candidate] =

copy(failure = Some(failure))

override def withResult(

result: Seq[ScoredCandidateResult[Candidate]]

): ScoringPipelineResult[Candidate] =

copy(result = Some(result))

}

object ScoringPipelineResult {

def empty[Candidate <: UniversalNoun[Any]]: ScoringPipelineResult[Candidate] =

ScoringPipelineResult(

None,

None,

None,

None,

None,

None,

None

)

}