package com.twitter.search.common.relevance.classifiers;

import com.google.common.base.Preconditions;

import com.google.common.collect.Lists;

import com.twitter.finagle.mtls.authentication.ServiceIdentifier;

import java.util.List;

import com.twitter.common\_internal.text.version.PenguinVersion;

import com.twitter.search.common.relevance.config.TweetProcessingConfig;

import com.twitter.search.common.relevance.entities.TwitterMessage;

/\*\*

\* Classifier that focuses on tweet text features and their corresponding

\* quality.

\*/

public class TweetTextClassifier extends TweetClassifier {

private TweetQualityFeatureExtractor featureExtractor = new TweetQualityFeatureExtractor();

private TweetTrendsExtractor trendsExtractor = null;

/\*\*

\* Constructor. Requires a list of TweetQualityEvaluator objects.

\* @param evaluators list of TweetQualityEvaluator objects responsible for quality evaluation.

\* @param serviceIdentifier The identifier of the calling service.

\* @param supportedPenguinVersions A list of supported penguin versions.

\*/

public TweetTextClassifier(

final Iterable<TweetEvaluator> evaluators,

ServiceIdentifier serviceIdentifier,

List<PenguinVersion> supportedPenguinVersions) {

Preconditions.checkNotNull(evaluators);

setQualityEvaluators(evaluators);

TweetProcessingConfig.init();

if (TweetProcessingConfig.getBool("extract\_trends", false)) {

trendsExtractor = new TweetTrendsExtractor(serviceIdentifier, supportedPenguinVersions);

}

}

/\*\*

\* Extract text features for the specified TwitterMessage.

\*

\* @param tweet TwitterMessage to extract features from.

\*/

@Override

protected void extractFeatures(TwitterMessage tweet) {

extractFeatures(Lists.newArrayList(tweet));

}

/\*\*

\* Extract text features for the specified list of TwitterMessages.

\*

\* @param tweets list of TwitterMessages to extract interesting features for

\*/

@Override

protected void extractFeatures(Iterable<TwitterMessage> tweets) {

Preconditions.checkNotNull(tweets);

for (TwitterMessage tweet : tweets) {

featureExtractor.extractTweetTextFeatures(tweet);

}

// Optionally try to annotate trends for all the tweets.

if (TweetProcessingConfig.getBool("extract\_trends", false) && trendsExtractor != null) {

trendsExtractor.extractTrends(tweets);

}

}

}