package com.twitter.search.common.util.ml.prediction\_engine;

import java.util.Arrays;

import com.google.common.base.Preconditions;

/\*\*

\* Represents a continuous feature that has been discretized into a set of disjoint ranges.

\*

\* Each range [a, b) is represented by the lower split point (a) and its associated weight.

\*/

class DiscretizedFeature {

protected final double[] splitPoints;

protected final double[] weights;

/\*\*

\* Creates an instance from a list of split points and their corresponding weights.

\*

\* @param splitPoints Lower values of the ranges. The first entry must be Double.NEGATIVE\_INFINITY

\* They must be sorted (in ascending order).

\* @param weights Weights for the splits.

\*/

protected DiscretizedFeature(double[] splitPoints, double[] weights) {

Preconditions.checkArgument(splitPoints.length == weights.length);

Preconditions.checkArgument(splitPoints.length > 1);

Preconditions.checkArgument(splitPoints[0] == Double.NEGATIVE\_INFINITY,

"First split point must be Double.NEGATIVE\_INFINITY");

this.splitPoints = splitPoints;

this.weights = weights;

}

public double getWeight(double value) {

// binarySearch returns (- insertionPoint - 1)

int index = Math.abs(Arrays.binarySearch(splitPoints, value) + 1) - 1;

return weights[index];

}

public boolean allValuesBelowThreshold(double minWeight) {

for (double weight : weights) {

if (Math.abs(weight) > minWeight) {

return false;

}

}

return true;

}

}