

Weka Decision Trees to Java Conversion

Egon Willighagen 

Published May 30, 2007

Citation

Willighagen, E. (2007, May 30). Weka Decision Trees to Java Conversion. *Chem-bla-ics*. <https://doi.org/10.59350/bnwdb-pa057>

Keywords

Java, Cheminf, Cdk

Abstract

Some time ago I wrote a small Perl script to convert a decision tree created with Weka in the ARFF format to Java source code, for use in the ionization potential prediction in CDK. The advantage is that Weka is no longer used at runtime, and that there is no model that needs to be loaded and interpreted. Instead, it is simple Java code that does the work, much faster.

Copyright

Copyright © Egon Willighagen 2007. Distributed under the terms of the [Creative Commons Attribution 4.0 International License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Some time ago I wrote a small Perl script to convert a decision tree created with [Weka](#) in the [ARFF format](#) to Java source code, for use in the [ionization potential prediction](#) in [CDK](#). The advantage is that Weka is no longer used at runtime, and that there is no model that needs to be loaded and interpreted. Instead, it is simple Java code that does the work, much faster.

This is the code:

```
#!/usr/bin/perl
#
# Copyright 2007 (C) Egon Willighagen
# License: GPL

use diagnostics;
use strict;

my $filename = $ARGV[0];

print "double result = 0.0;\n";
open(INPUT, "<$filename");
my $level = 0;
my $prevLevel = -1;
while (my $line = <INPUT>) {
    $line =~ s/\n//g;
    $level = 0;
    while ($line =~ /^\\|\\s*(.*)/) {
        $level++;
        $line = $1;
    }
    my $else = "";
    if ($prevLevel == $level) {
        $else = "else ";
    } elsif ($prevLevel < $level) {
        # we increase one level at a time
        for (my $i=0; $i<($level-1); $i++) { print " "; };
        print "{\n";
        $prevLevel = $level;
    } else {
        # this is a bit more tricky: we possibly need more than
        # one end bracket
        my $diff = $prevLevel - $level;
        for (my $closes=0; $closes<$diff; $closes++) {
            for (my $i=0; $i<($prevLevel-$closes-1); $i++) { print " "; };
            print "}\n";
        }
    }
}
```

chem-bla-ics

```
    $prevLevel = $level;
}
if ($line =~ /:/) {
    my ($if, $then) = split(":",$line);
    for (my $i=0; $i<$level; $i++) { print " "; };
    # FIXME: java-fy $then
    if ($then =~ /([\d|_]*)\s*\([([^\)]*)\])/) {
        my $result = $1;
        my $stats = $2;
        $result =~ s/_/\./g;
        print $else . "if ($if) { result = $result; // $stats }\n";
    } else {
        print $else . "if ($if) { result = $then; }\n";
    }
} else {
    for (my $i=0; $i<$level; $i++) { print " "; };
    print $else . "if ($line)\n";
}
}

# OK, now add the rest of the closing brackets
for (my $closes=$prevLevel; $closes>0; $closes--) {
    for (my $i=0; $i<($closes-1); $i++) { print " "; };
    print "}\n";
}
```