Algorithm 1: void buildClassifier(Instances dane)

```
atrybuty = pobierzAtrybuty(dane);
średnie[] = średnieDlaAtrybutów(dane);
buildTree(dane, atrybuty, średnie, depth=0);
```

Algorithm 2: void buildTree(Instances dane, int[] atrybuty, double[] średnie, int depth)

```
//K - liczba określajaca ile najlepszych par uwzgledniamy;
//uniquePairs - określa czy pary moga sie powtarzać;
if czyLiść(dane) then
    stwórzLiść();
end
pary = new List<>();
for i ← 1 to atrybuty.length do
    atrybut1 = atrybuty[i];
    for j ← i + 1 to atrybuty.length do
        atrybut2 = atrybuty[j];
        współczynnik = średnie[atrybut1] / średnie[atrybut2];
        int suma1, suma2 = 0;
        for x ← to dane.length do
            if dane[x].value(artybut1) > współczynnik * dane[x].value(artybut2) then
                if dane[x].classValue() then
                    suma1++;
                else
                    suma2++;
                end
            end
        end
        p1 = suma1 / dane[klasa1].length;
        p2 = suma2 / dane[klasa2].length;
        delta = Math.abs(p1 - p2);
        pary.add(delta, atrybut1, atrybut2, współczynnik);
    end
end
pary.sort(); // Sortowanie po delcie malejaco
najlepszePary = pary.wybierzNajlepsze(K, uniquePairs);
podzieloneDane[] = podzielDane(dane, najlepszePary);
średnie[] = średnieDlaAtrybutów(podzieloneDane);
for d ← 1 to podzieloneDane.length do
    buildTree(podzieloneDane[i], atrybuty, średnie, depth+1);
end
```