ID3 Algorithm

```
ID3(in T : table; C : classification attribute)
    return decision tree

{ if (T is empty) then return(null); /* Base case 0 */
  N := a new node;
  if (there are no predictive attributes in T) /* Base case 1 */
    then label N with most common value of C in T (deterministic tree)
                 or with frequencies of C in T (probabilistic tree)
  else if (all instances in T have the same value V of C) /* Base case 2 */
    then label N, "X.C=V with probability 1"
  else { for each attribute A in T compute AVG_ENTROPY(A,C,T);
         AS := the attribute for which AVG_ENTROPY(AS,C,T) is minimal;
         if (AVG_ENTROPY(AS,C,T) is not substantially smaller than ENTROPY(C,T)) /* Base case 3 */
           then label N with most common value of C in T (deterministic tree)
                        or with frequencies of C in T (probabilistic tree).
         else {
              label N with AS;
              for each value V of AS do {
                  N1 := ID3(SUBTABLE(T,A,V),C) /* Recursive call */
                  if (N1 != null) then make an arc from N to N1 labelled V;
                  }
              } }
  return N;
}


SUBTABLE(in T : table; A : predictive attribute; V : value) return table;
{ T1 := the set of instance X in T such that X.A = V;
  T1 := delete column A from T1;
  return T1
}


/* Note: in the textbook this is called I(p(v_1) ... p(v_k)) */
ENTROPY(in C : classification attribute; T : table) return real number;
{ for each value V of C, let p(V) := FREQUENCY(C,V,T);
  return -sum_V p(V) log_2(p(V)) /* By convention, we consider 0 · log_2(0) to be 0. */
}


/* Note; In the textbook this is called "Remainder(A)" */
AVG_ENTROPY(in A: predictive attribute; C : classification attribute; T : table)
  return real number;
{ return sum_V FREQUENCY(A,V,T) · ENTROPY(C,SUBTABLE(T,A,V)) }


FREQUENCY(in B : attribute; V : value; T : table) return real number;
{ return #{ X in T | X.B=V } / size(T); }
```