1. **sameclass(+Ns, ?C):** all the names in N are of class C.
2. **partition(+Ns, +A, ?NT, ?NF):** partitions the list Ns into two lists: NT and NF. NT has the names that have attribute A, and NF contains the names that don't.
3. **proportion(+Ns, +C, +A, ?P):** P is the proportion of elements of class C in the Ns list, for which an element A is true.
4. **entropy(+Ns, +As, +A, ?E):** E is the entropy of list Ns for an attribute A if it belongs to the list of attributes As, or 1.0 if it doesn't. The entropy for an attribute is calculated as: (−1) \* Σc∈clases p(c) \* log p(c). Where p(c) is the proportion mentioned previously. If p(c) is 0, then p(c) \* log p(c) is replaced by 0.
5. **minatr(+Ns, +As, ?M):** M is the attribute of the list As which has minimum entropy in Ns. If there are many possible values it returns the first in the list.
6. **maxcla(+Ns, ?C):** C is the most representative class of Ns.
7. **id3(+Ns, +As, +C, ?T),** T is the decision tree for a list of names Ns, a list of attributes As and class C. Each leaf of T is a leaf-node, leaf(C), where C is a class. Each non terminal node for an attribute A represents the class label of the final subset of this branch. It has the shape node(A, T1, T2) where A is an attribute and T1 and T2 are trees.
8. **tree(?T):** T is the tree we get from using id3 for the set of examples. The initial attribute list is the list of all the attributes. The initial class is the most representative of the training set examples.
9. **classify(+T, +N, ?C):** C is the class of the name according to a classifing tree T.
10. **clasification(Ns, ?Xs):** Xs is the list of clasifications obtained for the list Ns. Each element of Xs is a tuple (N,C) so that N is an element of Ns and C is a class.
11. **Call1(+F, -X):** F(X) e se esiste nel db da true.

La prossima volta dovremo capire come mergiare l’id3 con la somma\_pesata del prof che restituire la proporzione di ogni valore di un attributo e continuare dalla riga 92 dopo proportion(..).