Machine Learning *COMP 5/6630*

HW01: Decision trees

Remember that only PDF submissions are accepted.

1. How many decision trees are there with 3 binary attributes? With 4?

   The formula we can use for decision trees with n binary attributes:

   # of decision trees = 2^(2^n)

   Therefore, with 3 binary attributes the answer is 2^(2^3) = 256. With 4, the answer is 65536.

2. In class we looked at an example where all the attributes were binary (i.e., yes/no valued). Consider an example where instead of the attribute "Morning?", we had an attribute "Time" which specifies when the class begins.

   (a) We can pick a threshold τ and use (Time < τ)? as a criteria to split the data in two. Explain how you might pick the optimal value of τ.

   τ is divided in such a way that after the division, we can use a binary classification. For example, τ is 11:00, everything before or equal to that time goes to the left branch, everything after that time goes to the right.

   The way in which we would determine which τ is the best, is by using an information gain or loss function to figure out what the most efficient split would be. A good example of an algorithm that does this is the ID3 algorithm linked below.

   ID3 algorithm: https://en.wikipedia.org/wiki/ID3_algorithm

   (b) In the decision tree learning algorithm discussed in class, once an binary attribute is used, the subtrees do not need to consider it. Explain why when there are continuous attributes this may not be the case.

   With a binary attribute, the choice is either true or false. Once that question has been answered, asking the same question will only result in the same answer, and the question cannot be subdivided further either. Therefor that question is redundant.

   In the case of continuous variables however, this is not the case, because if you split a continuous variable, we do not end up with true or false. We end up with two continuous variables. Therefor we can revisit a question at any particular point in the tree, so that we can divide that data further into finer classifications.

3. Give two reasons why memorizing the training data and doing table lookups is a bad strategy for learning.
   - Memorizing training data and doing table lookup is in my eyes not machine learning and would only work if the table covers every possible combination of attributes. This is because it only works with test data that exactly matches the training data. if we encounter a case that is not in the table, then the algorithm will fail to give it a label.
   - Depending on the sort of data that is being memorized, the table can get very big. It would take a huge amount of time and storage to save all this information.