Exercise 4: TDT4171 Artificial Intelligence Methods

Jim Totland

March 2021

1 Decision Tree

a)

I have treated *Sex*, *Pclass* and *Embarked* as categorical variables. The resulting decision tree using these predictors is displayed in figure 1. We can see that the tree classifies females as survivors and men as non-survivors. It yields a test accuracy of $\approx$ 87.53%. To verify my code, I reproduced the restaurant-example from the lectures, which can be demonstrated by running `restaurantExample()` in the code.

Furthermore, I disregard *Name*, *Ticket* and *Cabin*, because I see no way to utilize these without modifying the data. I also disregard *Age*, because it has missing values. Hence, the relevant variables that I treat as continuous are *SibSp*, *Parch* and *Fare*.

b)

The resulting decision tree when including both categorical and continuous variables (as given in a) is shown in figure 2. It gives an accuracy of $\approx$ 88.73%

c)

We see that the test error rate is slightly higher in the purely categorical model from a) as opposed to the one from b) which includes continuous variables as well. However, the accuracy only increases by a little over one percent, which could seem a little modest, considering how much more information we include in b). Adding more predictors, however, might mean that predictors that are non-significant are used to explain noise in the training data and hence overfit the model to the training data. The modest increase in accuracy could be used as evidence that some of the continuous variables are irrelevant with respect to explaining the response.

One possible remedy for the overfitting would be to exclude splits on variables that do not provide enough information gain. This means that before we

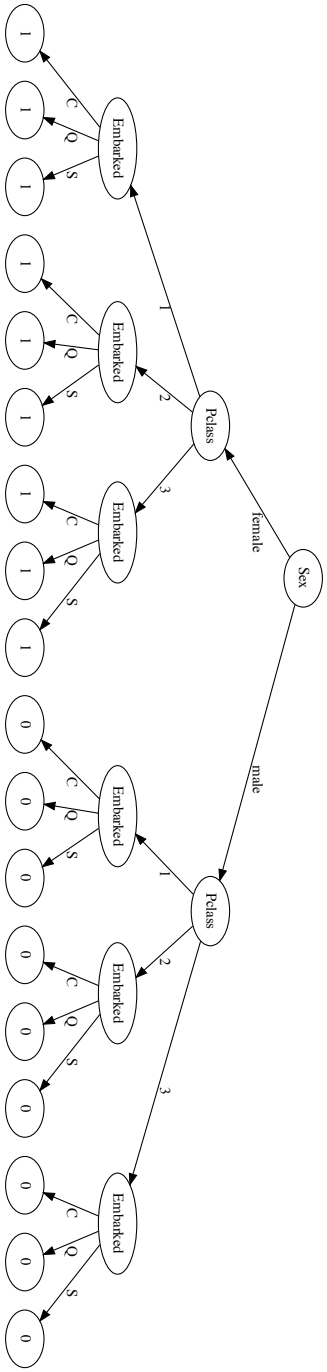

Figure 1: The decision tree with only categorical variables.

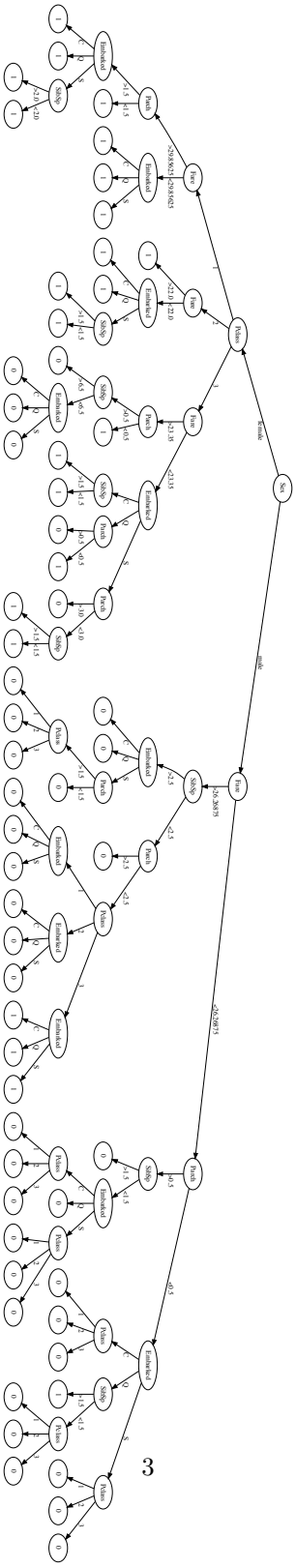

Figure 2: The decision tree with both continuous and categorical predictors.

split on a variable in the algorithm, we check if the information gain is below some threshold, and potentially stop if the split is not satisfactory. This could avoid splits on irrelevant attributes, hence reducing the risk of overfitting, and potentially improving the performance with respect to test accuracy.

This method does, however, have a significant drawback, because it would prune away branches where a 'poor' split occurs before a 'good' split. A better solution is to first build the full decision tree, and then successively prune the nodes with only leaf-nodes as children using a χ^2 statistic as described in the course book. This could also reduce overfitting and hence improve accuracy on test data.

2 Missing Values

One way of dealing with missing data when splitting on an attribute A in the training phase is to try classifying all the examples with unknown value for A to each of the categories and then calculate the resulting information gain. The classification of the unknowns which yields the highest information gain is then chosen as the classification rule for unknown values at that node. For continuous variables, we first calculate the optimal split based on the known values (and information gain) and then try classifying the unknowns to either side of the split. This is relatively simple to implement, but it could also be a too simple classification rule. E.g. if we have two examples with unknown values for Age, which in reality are $Age_1 = 95$ and $Age_2 = 2$, they would be classified to the same category. This might be an unreasonable simplification.

A method that can be used in the testing phase: For an attribute A, assign the mean (or majority class if A is categorical) of the known values in the test examples to all the unknown values. This is also very easy to implement, and could be a reasonable approach if there are few unknown values. On the other hand, if the amount of unknowns is significant, this approach may be too crude.