NTNU
Kunnskap for en bedre verden

TDT4171 - Artificial intelligence methods

Assignment 4 - Decision trees

Sander Lindberg

March, 2021

1 Decision trees

Task 1

Problem a

In this subtask, I decided to split only on the attributes "Sex", "Pclass" and "Embarked", as we were only supposed to split on the categorical variables. I debated with myself whether or not "Parch" and "SibSp" could pass as categorical variables, since a person could be placed in e.g. the categories "has 2 siblings/spouses aboard the Titanic" and "has 1 parent/child aboard the Titanic". However, I concluded that these attributes were continuous since, theoretically, you could have X siblings/spouses/children aboard the Titanic, compared to the limited Pclass 1,2,3, Embarked S, Q and C and the two genders Male and Female. My decision tree can be seen in figure 1 below.

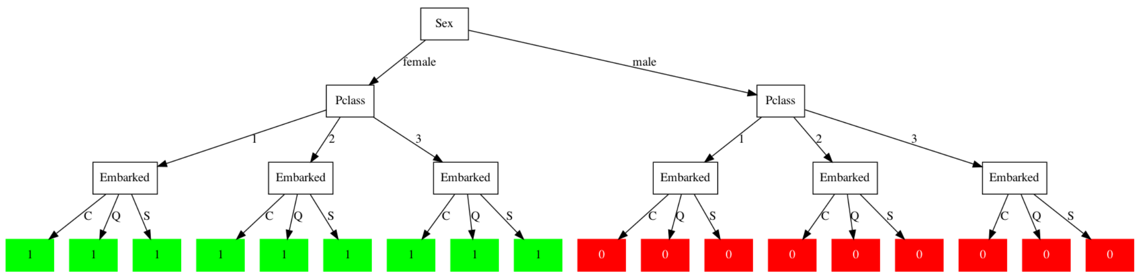

Figure 1: Caption

I got an accuracy of 0.8752997601918465 when testing with the test set. The way I calculated the accuracy was to loop though each row in the dataset, and then traverse the tree generated by *Decision-tree-learning* until I reached a leaf-node. The value this node holds will be the predicted survived value for the row I'm currently at. I then compared this value with the actual survival value and added the result to a list. After this was done for all rows in the dataset, I summed up all values and divided by the length of the prediction list. This gave me the accuracy $acc = \frac{Tp+Tn}{Tp+Fp+Tn+Fn}$. The accuracy calculation can be seen below.

```python
def predict(tree, data):
    if not len(list(tree.branches.items())):
        return tree.value
    return predict(tree.branches[data[tree.label]], data)

predictions = []
test = pd.read_csv('titanic/test.csv')
for ind, row in test.iterrows():
    prediction = predict(tree, row)
    predictions.append(prediction == row['Survived'])

print(f'Accuracy: {sum(predictions)/len(predictions)}')
```

Problem b

I even though the column "Age" has missing values, I decided to add it as a continuous variable, since I got a slightly higher accuracy when including it. The way I handled the missing values was to simply fill them with the mean of all age-values (data. fillna (data.mean(), inplace=True)). Other than that, I included "Parch" and "SibSp" as continuous variables. The only reason I did not include "Ticket" is because I do not think it is a suitable predictor. I did not include "cabin" because it contains noise (see problem C). This, together with my implementation that supports continuous variables gave me an accuracy of 0.8872901678657075, which is slightly higher than the accuracy in task a (+0.012) and the decision tree seen in figure 2.

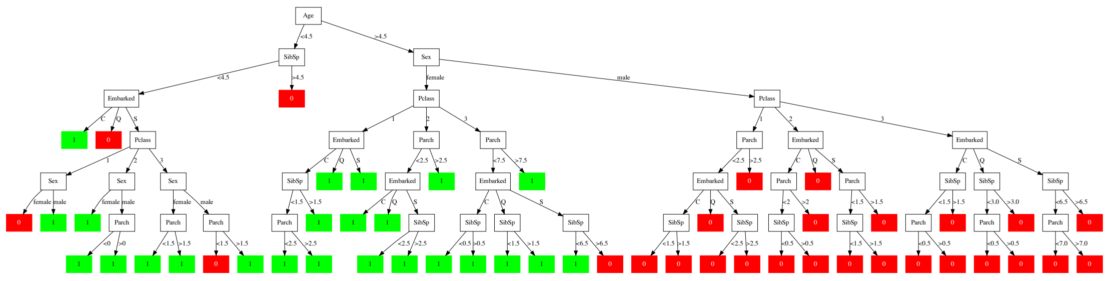

Figure 2: Decision tree for task 1b

Problem c

My results from task a and task b are quite similar. When adding support for continuous variables, the accuracy was increased by 0.012 or 1.2%. I expected the accuracy to increase a little bit, since, after manually exploring the dataset, I could see that not all females survived, as the tree in task a suggests. Adding more attributes gives more ”evidence” and a better basis for prediction. On the other hand, adding more attributes may lead to more ”guess-predictions”, like we do not have this exact data in the trainset, so the algorithm has to guess a value for survived. This guess may be wrong and will lead to lower accuracy.

A technique that *could* improve accuracy is pruning. Pruning helps eliminate overfitting and removes irrelevant attributes. This can, however, still lead to the problem described above, where the guess are wrong.

Another thing I would have done that *could* have improved accuracy on this exact dataset is to clean the ”cabin” attribute and include it in the tree (as a categorical variable). Most of the cabin-data are stored as e.g. ”B57”, a letter followed by a number. Some of the cabins, however, are ”B57 B59 B63 B66” and ”F E69”, which could be disturbing for the model. What I would have done is to clean this data to only contain the letter (A, B, C, etc. . .) and used them as categorical variables. This is because I believe cabins with the same letter are located close to each other, and it would have been interesting to look at how and if the survival rate of e.g. cabins A and F differs from each other. (Believing that some cabin letters may be placed further from e.g. life boats than other.)

Task 2

One method to handle missing data is to simply set the missing values to the mean of the rest of the attribute’s values. One advantage using this method is that it can prevent data loss, as opposed to simply dropping the whole column as we (should) have done in this assignment. A disadvantage, however, is that we add a lot of approximate values, especially if the count of the missing values are big, possibly adding noise to the model. One assumption I am making by suggesting this method is that the values are actually missing, and not just not included for some reason. E.g. every person *has* an age, it is just missing from this dataset for some reason.

Another method to cope with the missing values in the dataset is to remove the rows with the missing values. An advantage with this method is that we do not approximate the missing and we can still keep the attribute as a classifier, possibly making our model predict better. A disadvantage is that we miss out on valuable data from the other attributes the row we are deleting contains. An assumption that is crucial to take when using this method is that you have enough data to be able to delete rows. E.g. if you have 100 rows of training data and you delete 50 of them due to missing values, you get a lot less ”evidence” in your model compared to deleting 50 out of 1000 rows.