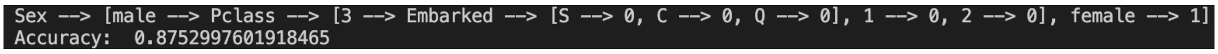
TDT4171 – Assignment 4

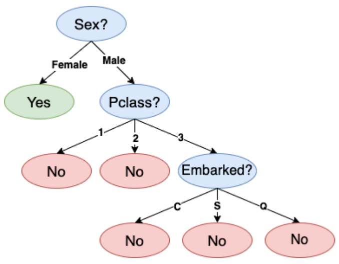
The continuous attributes in the dataset are **SibSb** (number of siblings/spouses), **Parch** (number of Parents/Children) and **Age**. Because age has missing values it is not used in the implementation for continuous values in b). Attributes such as **Name**, **Ticket**, **Fare** and **Cabin** are left out because splitting on these attributes would not make sense. This is the case because e.g., every passenger has a unique ticket or name, and a splitting on this attribute would lead to a huge tree containing no usable information when predicting unseen samples. Even tough **Cabin** isn't unique for every example, splitting on this attribute would also lead to a huge tree containing little/no information when predicting unseen examples. In both task a) and b) the structure for storing the tree is adapted from [1].

Task 1:

a)

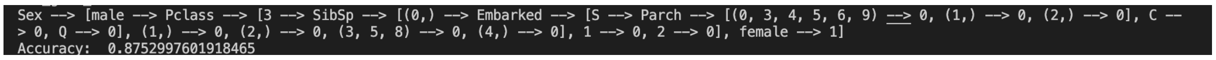


```
Sex --> [male --> Pclass --> [3 --> Embarked --> [S --> 0, C --> 0, Q --> 0], 1 --> 0, 2 --> 0], female --> 1]
Accuracy:  0.8752997601918465
```

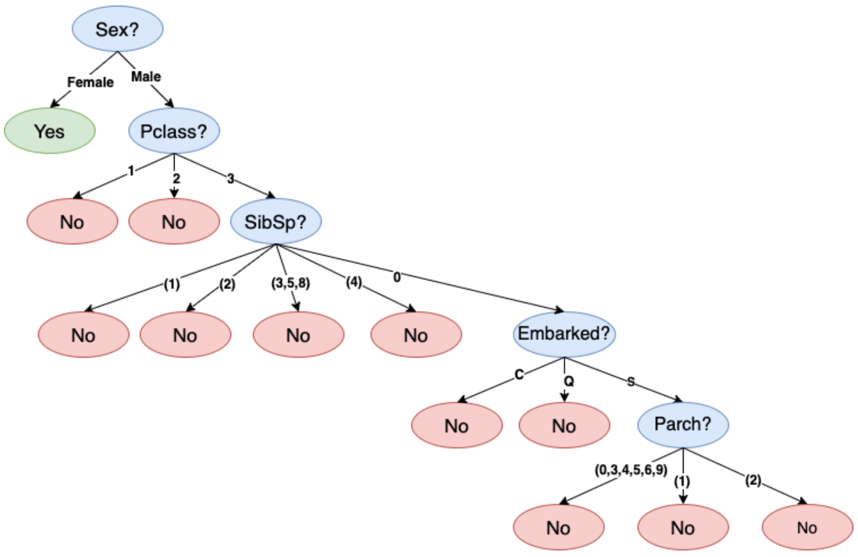


b)

The algorithm assumes that attributes with more than 3 values are continuous. In the importance these values are split into all possible partitions with size greater than one and smaller than len(values)-1. The partition one with maximum gain is returned and compared with the other attributes. The method for partitioning the values is adapted from [2].



```
Sex --> [male --> Pclass --> [3 --> SibSp --> [(0,) --> Embarked --> [S --> Parch --> [(0, 3, 4, 5, 6, 9) --> 0, (1,) --> 0, (2,) --> 0], C --
> 0, Q --> 0], (1,) --> 0, (2,) --> 0, (3, 5, 8) --> 0, (4,) --> 0], 1 --> 0, 2 --> 0], female --> 1]
Accuracy:  0.8752997601918465
```



c)

The accuracy of the implantation with- and without continuous attributes are the same. I expected the accuracy to be a little higher for the implementation including continuous attributes, as we have more information available.

To get a better performance one solution would naturally be to include **Age** as an attribute. Another method for splitting this attribute would then be necessary, as it makes more sense to split the age based on intervals (i.e., age < 10). This is likely to improve the performance because we would have a lot more information.

In the implementation from b) the continuous variables are split independent of ordering. Another solution might be to split all the continuous attributes based on intervals, using the same method mentioned above (for **Age**). This might improve the performance on the test-set because the method testing all possible partitions may lead to overfitting. This method of splitting could also be used for **Cabin**, and the accuracy might improve if there are some correlation between cabin-number-interval and survival.

Task 2:

One solution would be to discard all examples with missing values for the attributes that we are going to split on. One disadvantage for this method is that we may lose valuable information, e.g., if the examples with missing have other attributes in common. This could for instance be that passengers traveling on third class are more likely to have missing values for age, and we would then loose a lot of information for this group. The advantage of this method is that it is easy to do. Another method might be to predict the missing value based on the other attributes.

References

[1] https://philippmuens.com/decision-trees-from-scratch
[2] https://stackoverflow.com/questions/19368375/set-partitions-in-python/30134039#30134039