TDT4171 Artificial Intelligence Methods

Decision Trees

March 4, 2021

- **Delivery deadline: March 17, 2021 by 23:59.**
- Required reading for this assignment: Chapter 18 (the parts in the curriculum).
- Deliver your solution on Blackboard. Please upload your report as a **PDF file**. For the programming part, please upload the source code alongside the PDF. Please **do not** put the PDF or source code in an archive.
- Students can NOT work in groups. Each student can only submit solution individually.
- This homework counts for 10% of the final grade.
- This assignment totals 10 points. The number of points for each problem is labeled.
- Cribbing("koking") from other students is not accepted, and if detected, will lead to immediate failure of the course. The consequences will apply to both the source and the one cribbing.

1 Decision Trees

8 points

In this exercise you are to implement a decision tree classifier. You are **not** allowed to use an implementation of decision tree classifiers from libraries such as sklearn. You are **not** allowed to copy code you have found online. **Read the entire exercise before writing any code or answering any questions.**

a) **4 points** Implement a decision tree that can support categorical variables. This algorithm is described in Figure 18.5 in AIMA. Use *information gain* as the IMPORTANCE function. Use your implementation to train a decision tree on the Titanic dataset (see below). Some of the columns in the dataset are continuous attributes, and cannot be used with this implementation. Which are these? Test your model on the test set and print your accuracy. Add your decision tree to the PDF manually, using Graphviz, or with some other software.

b) **2 points** Extend your implementation to also support continuous variables. Train this model on the Titanic dataset again, now also using the continuous attributes you left out in **a)**. Test the model again on the test set and print the accuracy of the predictions. Add your decision tree to the PDF manually, using Graphviz, or with some other software.

c) **2 points** Discuss the results from **a)** and **b)**; is the performance difference what you expected? What might be done to improve the performance of the decision tree classifier on the test set? Suggest at least 2 changes that might be made to the algorithm that could improve performance. Argue why your changes *could* improve performance. You do not have to show that it actually does improve accuracy on the Titanic dataset.

Continous variables

Splitting on continuous variables is different from categorical variables. You cannot make a branch for all possible values, instead we find the best value to split on and make two branches, one for numbers smaller than the split, and one for numbers larger than the split. Because you are only making two branches at each split for continuous attributes, typically we let the decision tree split on those attributes multiple times. For your implementation in **b)** we only require you to be able to split on the continuous attributes *once*, just like with the categorical attributes. This makes it so that the tree will never become deeper than the number of attributes in the data.

Titanic dataset

The Titanic dataset contains information on passengers aboard RMS Titanic. The idea is to predict the survival of each passenger based on other collected

information.

The dataset contains the following columns:

Survival:	Did the passenger survive (0=No, 1=Yes)
Pclass:	Ticket class (1=1st, 2=2nd, 3=3rd)
Name:	The name of the passenger
Sex:	Sex
Age:	Age in years
SibSp:	Number of siblings/spouses aboard the Titanic
Parch:	Number of parents/children aboard the Titanic
Ticket:	Ticket number
Fare:	Passenger fare
Cabin:	Cabin number
Embarked:	Port of embarkation (C=Cherbourg, Q=Queenstown, S=Southampton)

Not all columns are suitable predictors, however. Use only the columns you find suitable for predicting survival rate and explain why you disregard the other attributes.

Data is split into two files:

- Training set (train.csv)
- Test set (test.csv)

The training and test sets includes all columns listed above. Furthermore, some of the columns contain missing values. You do not need to implement support for missing values, so you can disregard these attributes when training your decision tree.

We have given you a modified version of the Titanic dataset available at `https://www.kaggle.com/c/titanic/data`.

Tips

Use a dataframe library to load the csv files. Pandas is a great option if you are using Python. Other Pandas methods that can come in handy when implementing the decision tree include *DataFrame.value_counts*, *DataFrame.groupby*, *DataFrame.map*, and *DataFrame.sort_values*.

2 Missing Values

2 points

One of the columns in the Titanic dataset contained missing values. We chose to not use these columns at all when we implemented out decision tree, even

though only very few values were missing. This means that we missed out on a lot of information that could have made our decision tree perform. Suggest at least two methods that we could have used to handle missing data either during training or prediction. Describe your proposed solutions either in text or with pseudo code. Discuss the advantages and disadvantages of your methods. What assumptions are you making?

3 Solutions

Decision Tree

a) Continuous attributes are *Age*, *Fare*, *SibSp*, *Parch*. Can only use *Pclass*, *Sex* and *Embarked*.

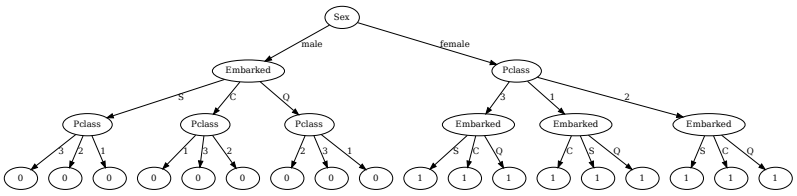
b) See python code.

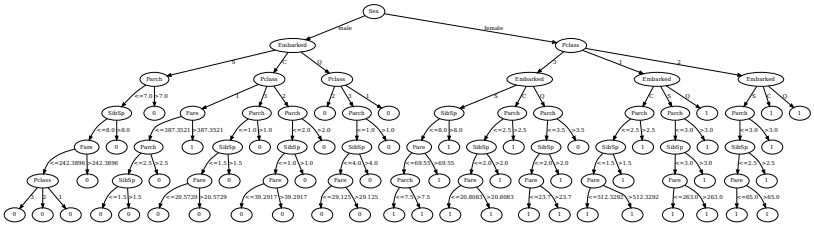
c) We see almost no improvement when including the new attributes. This can be because they are not good predictors, or because the information they contain is already present in the categorical attributes. When given sex, no other attributes seem to have any predictive power.

Missing Values

Here are some methods that can be used to handle missing data

- Discard all columns with missing values during training. This assumes that the distribution of missing values is completely random. Otherwise we may have discarded some datapoints with more frequency than others.
- Train a decision tree to predict the missing value based on the other attributes. Use this decision tree whenever you have to split on a missing value. This assumes that the missing values are dependent on the other observable attributes (but not the value of the missing data itself).
- For prediction you could follow all subtrees during a split on a missing value, and let the prediction be the (weighted) mean prediction across all subtrees. The weighing would be relative to the information/number of datapoints in each subtree




Sex
male
female
Embarked
Pclass
Parch
Pclass
Pclass
Embarked
Embarked
Embarked
SibSp
Fare
Parch
Parch
Parch
Parch
SibSp
Parch
Parch
Parch
Parch
Parch
Fare
SibSp
SibSp
SibSp
Fare
SibSp
SibSp
SibSp
Fare
SibSp
SibSp
SibSp
Pclass
SibSp
Fare
Fare
Fare
Fare
Parch
Fare
Fare
Fare
Fare
Fare
Fare
Fare
Fare
Fare
Fare
Fare