CSC421: Artificial Intelligence
Assignment 4

Rafael Solorzano

June 27, 2018

1 Introduction

In this assignment logic programming using logpy as well as a standard python is performed for a family tree logic. Further. Hidden Markov Models in python using hmmlearn is performed, to examine different states of healthy or unhealthy as well as moves performed according to this hidden states.

2 Logic Programming

2.1 logpy

Logic programming to determine parents, grandparents and children of different Star Wars characters was performed using logpy. First, the parent relationship is created as facts in which we state (”Parent”, ”Children”) the relationships created for this problem is shown in Figure 1. Once the relationships have been created we can query direct results from them, we query ”Who is the parent of Luke Skywalker” as well as ”Who are the children of Darth Vader” the code for this queries is shown in Figure 2. Finally, in order to query ”Who is the grandparent of Kylo Ren?” we first create a an auxiliary grandparent function, and then we query using this new function, the code for this is shown in Figure 3. The results of all these functions is shown in Figure 4.

2.2 Standard Python

The same problem with the same queries is also performed using Standard Python. We use a dictionary of lists to establish the facts, this is created in the form ”Parent” : [”Child 1” ... ”Child n”]. Storing the facts this way, allows us to query for direct children, as well as using a function we can query for every parent of a child, by checking if a child is listed under a parent. Finally, in order to perform the grandparent check, we get the parents of the grandson first, further we get the parents of the parents of the grandson, in the same manner as in parent function. The entire code using Standard Python is shown in Figure 5, and its result is shown in Figure 6.

Contrasting the two methods, the logic programming allows for easier and faster queries as well as more robust results. Using logpy or Prolog allow for easy and robust logic programming, with very strong query processing. Whereas, with standard python using dictionaries we have to define every single function of what we want to achieve, and provide precise results of how to do this. For this particular problem as it is very small set, doing it wit logpy yielded with a lot less code, and only one extra function being needed, whereas for standard Python we required a separate function for each problem that we later can call to get the results.

3 Hidden Markov Models

In order to generate random sequence of healthy, and injured an approach similar to the card deck in Monopoly is used. A list of different states labeled 1 for Healthy and 0 for injured is created. For example the healthy matrix would contain 7 1's and 3 0's to represent 0.7 probability of staying healthy and 0.3 of going injured. Further, to generate random sequences of dribble, pass, or shoot given the current state is doing in the same manner as for healthy injured. The code for the generation of this sequences is shown in Figure 7, whereas a sample printout of each function is shown in Figure 8.

After we have created the sequence of observations, we create a Hidden Markov Model using hmmlearn with Multinomial Hidden Markov Model. First, we create the model object to which we assign the initial probabilities, this are 1.0 and 0.0 as the initial state is always healthy not chosen by random. Second, we define the transition matrix from this states. Third, the emission probability of the different moves is specified. The code for the creation of this model is shown in Figure 9.

Once we have the model we can fit, the generated sequence of moves into our model. After fitting the we fit the model to this sequence of observations. The code of fit and the printout of its results is shown in Figure 10 and 11 respectively.

The predict function of the HMM model will predict the sequence of hidden states, that was used to generate our sequence of observations. It was decided to run this predict function 10 times with different observations and average the results in order to provide how many errors the predict function produced. The code that does the predict is shown in Figure 12. The results obtained for each run were:

With an average of X errors.