CSC421 Assignment 4. Spring 2006 (10pts)

If an expert system –brilliantly designed, engineered and implemented– cannot learn not to repeat its mistakes, it is not as intelligent as a worm or a sea anemone or a kitten. From The Gardens of Learning, AI Magazine 14(2) (Summer 1993).

Selfridge, Oliver G.

`Student Name:` `Student Number:`
`Instructor George Tzanetakis`

Question	Value	Mark
1	3	
2	3	
3	4	
Total	10	
Extra Credt	3	

1 Overview

The goal of this assignment is to familiarize you with Hidden Markov Models (HMM), Decision Trees, and Statistical Learning. Be aware that some of the questions require much more work than others. Your deliverable will be a report documenting your answers. **Clarity and correct use of mathematical notation will count in grading**.

You can use any programming language to implement the programming parts of the assignment. Don't hesitate to contact the instructor via email or the anonymous comment form with any questions/clarifications you might need.

2 Probability Theory (3pts)

After inventing a time machine, in your free time between CSC421 assignments, you find your self stranded sometime in the middle ages. After trivial details irrelevant to this assignment such as surviving, learning the language and realizing that your time machine is broken you find yourself appointed as the scientific advisor to some king and are given the task of decrypting the coded messages of his enemies captured from carrier pigeons. The king has spies who have reported that his enemies use the following process to encrypt the messages. Each letter of the alphabet is mapped to another letter deterministically (for example A becomes G , B becomes H, I becomes J etc) but they don't know the mapping. Given that your computational resources are limited (no computers in the middle ages but you have 20 monks to help out) you set out to break the ciphertext. You set the monks to count letter frequencies in the language and armed with that and a little bit experimentation you happily break the cipher.

Unfortunately when the enemies find out that their code has been broken they improve their procedure in the following way. Each letter of the alphabet is mapped to **any of three letters with equal probability. The letters belong to a larger alphabet such that the frequency counts of each letter in the cipher text are approximately equal.**

*Describe how you and your monks can break this new scheme (hint: think about First-Order Markov letter transitions) (***1pt***) ?*

Unfortunately, again the enemies improve their procedure in the and the spies bring the following information. For each letter of the alphabet that needs to be encoded they roll a 6-face unbiased die six times and note down the sum of all the numbers. Then they consult a notebook that for each letter contains a table with substitution entries for each possible value of the sum. For example, an entry for encoding A might be (6,Z) meaning that if the sum of the six dices is 6 then encode the letter A as Z or (20,B) meaning if the sum of the six dices is 20 then encode the letter A as B. There is a separate table for each letter of the alphabet in the notebook.

*Formulate this process as a Hidden Markov Model using the mathematical notation conventions we have learned (***1pt***).*

*Describe how you and your monks can try to break this improved scheme and sketch a simple hypothetical example of decoding (no need to provide all the numbers and calculations - but show enough so that the other advisors of the King who have not taken CSC421 understand the process) (***1pt***).*

3 Learning and Decision Trees (3pts)

Choose any application area that you like and create a training set of of attribute-value entries with discrete values and a positive/negative label (similar to the restaurant example in your book). Your data should have a minimum of 10 positive entries, 10 negative entries and each entry should have a minimum of 4 attributes. You can either get the values using your imagination and reasonable assumptions or you can obtain them from actual examples (for example by asking your friends questions or by observing the weather etc).

- Draw the full trivial decision tree based on the truth table. **(1pt)**
- Using the information-gain heuristic and your training set show how a simpler decision tree can be learned. **(1pt)**
- Write your training set as a Weka attribute file (.arff) and load it into the Weka explorer interface. Run the ID3 classifier and include the decision tree and evaluation output given by Weka in your report. `http://www.cs.waikato.ac.nz/ml/weka/` **(1pt)**

4 Statistical Learning (4pts)

1. (**1pt**) Most programming environments provide a function that returns uniformly-distributed random numbers in some range. Frequently you are required to generate random numbers that follow other distributions. The exponential probability distribution function is defined as:

   $$p(x) = \theta e^{-\theta x} \tag{1}$$

   Plot the exponential distribution function for $0 < x < 100$ and $\theta = 0.1$. Write a function *randExp* that takes as arguments θ and n (number of samples) and returns n random samples of the corresponding exponential distribution (hint: you need to invert the pdf). Make a plot of 100 random samples with θ equal to 0.1. The mean and standard deviation of an exponential distribution are both equal to $\frac{1}{\theta}$. Show that the mean and standard deviation of your generated samples is the same.

2. (**1pt**) Create a histogram of your exponentially distributed random samples with 100 bins such that the sum of bars is 1.0. Plot the histogram for 10^2, 10^3 and 10^4 random samples. Confirm that the resulting histograms approximate the original pdf plot.

3. (**2pt**)

   In this part we are going to explore statistical machine learning. You need to write a function to generate a training set represented as a matrix where each row is $[x, y, l]$, where x and y are your features and l is your class label (1 or 2).

   To generate the training samples you will utilize the *randExp* function you wrote above. Class 1 should contain (x,y,1) points , where x is exponentially distributed with mean value of 20.0 and y is exponentially distributed with mean value of 15.0. Similarly class 2 should contain (x,y,2) points, where x is exponential distributed with mean value of 30.0 and y is exponentially distributed with mean value of 30.0.

   Create a training set with 50 examples for each class. Make a scatter plot of the resulting training set (the points are put on a plane based on their (x,y) coordinates with different symbol/color for each class).

   Calculate the Bayes classification error (the "lowest" possible) by estimating the class of each point in your training set using the likelihoods calculated from using the exponential distributions used to create the training set. (Because we created the training set we can know the "best" answer). How many errors are made by this process ?

   Now, assume that you model the class dependent probability density functions using a multivariate Gaussian classifier with parameters estimated from the training set (basically the empirical mean and standard deviation for each feature). You can assume that the features are statistically independent. Calculate the classification error by estimating the class of each point in your training set using the likelihood using the Gaussian classifier. How many errors are made by this process ?

5 (Extra Credit) HMM encoding/decoding

In any programming language you like, implement the encoding/decoding schemes of question 1 **(2pt)**.

Use at least 4 classifiers in Weka to calculate classification errors for the dataset of question 3 and report on your comparison. **(1pt)**.

6 Deliverables

Your deliverable is a report with your answers to the questions. For the questions requiring programming you must include the source code and test cases and provide enough documentation to make it easy to understand and read.