Assignment 3

CMSC 471 (03/01) — Artificial Intelligence

Item	Summary
Assigned	Sunday April 4th
Due	Wednesday April 14th, 11:59 PM Baltimore time
Topic	FOL & Machine Learning Evaluation
Points	80

In this assignment you will gain experience with first order logic and some basic machine learning evaluation techniques.

You are to *complete* this assignment on your own: that is, the code and writeup you submit must be entirely your own. However, you may discuss the assignment at a high level with other students or on the discussion board. Note at the top of your assignment who you discussed this with or what resources you used (beyond course staff, any course materials, or public Piazza discussions).

The following table gives the overall point breakdown for this assignment.

Question	1	2	3	4
Points	15	25	15	25

What To Turn In You must turn in two items:

1. A writeup in PDF format that answer the questions.
2. A single `zip` or `tar.gz` file containing all code, environment files (if applicable) and execution instructions necessary to replicate your output.

As part of your submission, be sure to include specific instructions on how to build (compile) your code. Answers to the following questions should be long-form. Provide any necessary analyses and discussion of your results.

How To Submit Submit the assignment on the submission site:

https://www.csee.umbc.edu/courses/undergraduate/471/spring21/01_03/submit.

Be sure to select "`Assignment 3`."

Questions

1. (**15 points**) Translate the following sentences into first order logic (FOL). If it's easier, you may reinterpret each sentence prior to translation into FOL.

   (a) "Every friend of Picard's is a friend of mine."

   For this, use the predicates `friend(x, y)` to mean "x is a friend of y," and the constants `Picard` and `Me`.

   (b) "Even cats who like to be pet can be overstimulated."

   For this, use the predicates `cat(x)`, `likes_petting(x)`, and `overstimulated(x)`.

   (c) "No CS student should know only one programming language."

   For this, use the predicates `CS_student(x)`, `know(x,y)`, and `pl(x)` (programming language).

2. (**25 points**)

   (a) Write a collection of FOL rules to encode the necessary and sufficient conditions of two people `x` and `y` being "Mth cousins." That is, find an appropriate expression for ". . ." such that the following defines `x` and `y` being Mth cousins:

   $$\texttt{cousins}(x, y, m) \leftrightarrow \ldots$$

   You may assume x and y are variables referring to people, while m is a positive integer.

   We say that `x` and `y` are "Mth cousins" if they have a common ancestor $M + 1$ generations in the past. For example, two people are first ($M = 1$) cousins if they have the same grandparent, and two people are second cousins ($M = 2$) if they have the same great-grandparent. Note that if $\texttt{cousins}(x, y, m)$ is true, then $\texttt{cousins}(y, x, m)$ is as well (it's symmetric).

   You won't be able to do this in one single rule. You will need to write a collection of rules for this. Use the predicate `parent_of(x, y)` (x is the parent of y). If you need to, you can define the predicates as you need, but be sure to explain them.

   My hint: start small. Try to write a specialized rule for first cousins (e..g., $\texttt{first\_cousins}(x, y) \leftrightarrow \ldots$), then for second cousins, and then generalize. You don't have to, but you'll probably find it much easier!

   (b) Based on the (limited) British Royal family tree (Fig. 1), use resolution and the rules you wrote in part (a) to:

   (i) *Prove* that William, Duke of Cambridge and Peter Phillips are first cousins.

   (ii) *Prove* that William, Duke of Cambridge and Harry, Duke of Sussex are *not* first cousins.

   (iii) Prove *or* disprove whether Archie Harrison and Lena Elizabeth are third cousins.

   To answer this question, you'll need some "facts." Assume only `parent_of` facts have been instantiated.

3. (**15 points**) Describe a situation (ML problem) where you want to maximize each score (even at the possible expense of the others).

   (a) Accuracy

   (b) Recall

   (c) Precision

   (d) F1

4. (**25 points**) This question is concerned with computing recall, precision, F1, and accuracy scores based on a list of predicted values and the corresponding correct values. For each subpart that requires computation, provide the contingency table (the values of TP, FP, FN, TN), and the recall, precision, F1, and accuracy scores.

   (a) Produce the contingency table and four scores for a binary classification result, where the correct labels are
   `[T, T, F, T, F, T, F, T]` and the predicted labels are
   `[T, F, T, T, F, F, F, T]`. Assume `T` means "true" (the desired class: the one you're trying to predict) and `F` ("false") is the "default" class.

   (b) Write code to compute recall, precision, F1, and accuracy from a list of predicted values and the corresponding correct values. There is no starter code for this question: just describe how to run your code.

      (i) Verify your results from part (a), and provide a screenshot of this/the output from your code.

      (ii) Compute the four scores for the predictions and correct values contained in the file
      `https://www.csee.umbc.edu/courses/undergraduate/471/spring21/01_03/materials/a3/predicted_actual.csv`
      Provide a screenshot of this/the output from your code.
      This is a two column CSV file (with a header). Each row (after the header) is a different instance: the first column lists the actual label, while the second column lists the predicted label. For this file, you're trying to predict the "1" class (so "0" is the default class).

   (c) Use the sklearn metrics module (`sklearn.metrics`) to verify your results from (b.i) and (b.ii). For this question,

      (i) provide the function call(s) you need to make in your writeup.

      (ii) for every function argument, except for `y_true` and `y_pred`, you use in the calls to the sklearn functions, explain *why* you need them.

      As a hint, remember which label you're trying to predict!

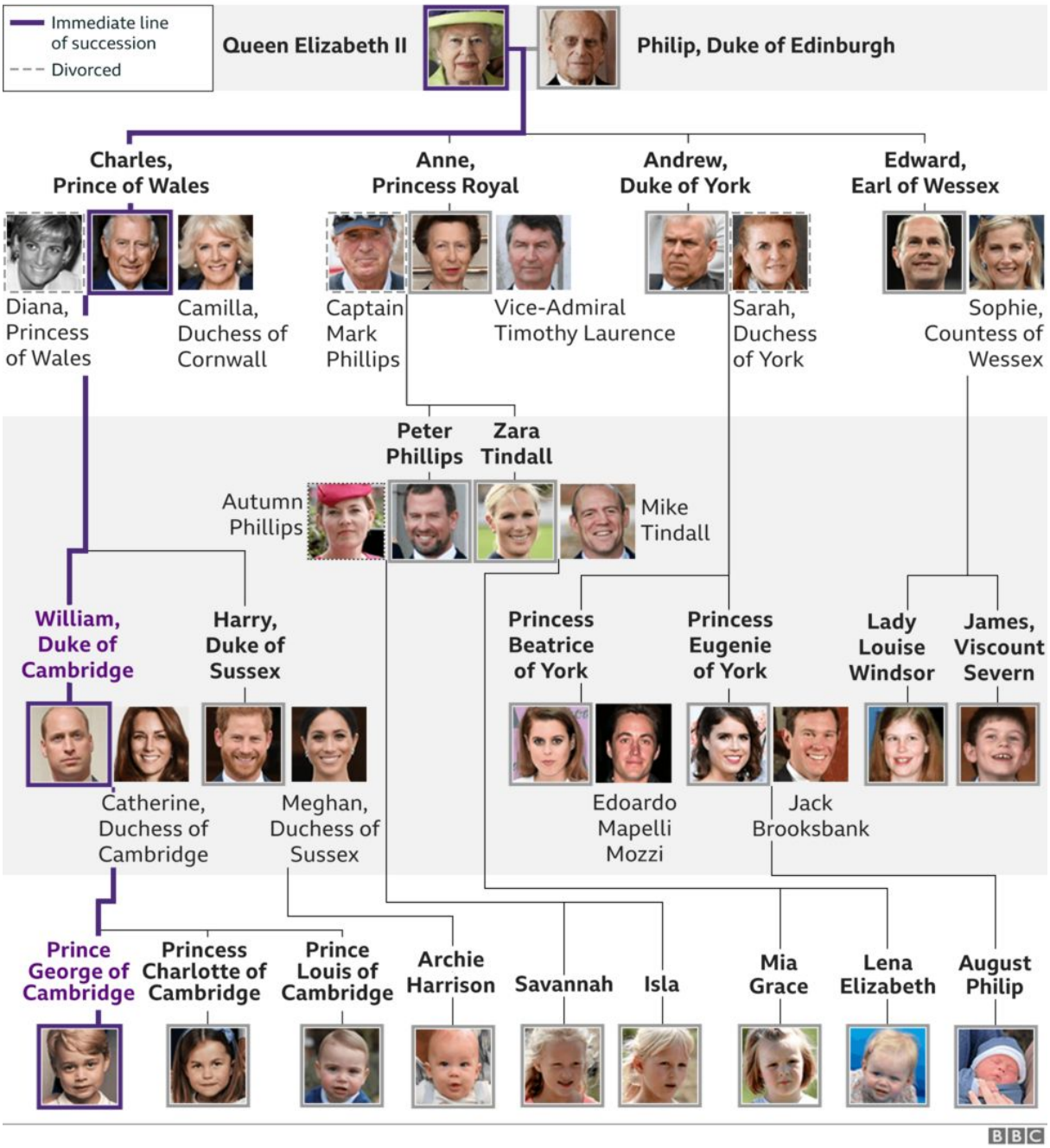

Figure 1: Royal family tree, from `https://www.bbc.com/news/uk-23272491`