CS440
Assignment 5
Due April 26, 2019

Computer Science Department
Colorado State University

April 18, 2019

Preliminaries. You are going to experiment with random forests (decision trees). The goal is to predict whether two RNA sequences interact. Hence, the input is two RNA sequences. The output is positive/negative. In this assignment, you will consider three different representations of input: (i) ACGT alphabet, (ii) one-hot kmers vector, and (iii) DeepBind matrix.

Training data. You are given six data files that you are going to use for training your model:

- `seq_negative_training.csv`
- `seq_positive_training.csv`
- `kmer_negative_training.csv`
- `kmer_positive_training.csv`
- `deepbind_negative_training.csv`
- `deepbind_positive_training.csv`

Raw sequence representation. In this representation, the samples are in a CSV file with 4 columns: (i) sequence of the first RNA, (ii) sequence of the second RNA, (iii) length of the first RNA, (iv) length of the second RNA. Notice that for some of the samples one or two of the sequences are empty strings. You should exclude them when training a model. Notice that in this representation the labels are not added to the data; positive and negative data are in separate files.

One-hot k-mer vector representation. A substring of length k is called a k-mer. In this representation, we consider all the possible k-mers that can be generated by the 4 characters A, C, G, and T in lexicographic order and make a vector of length 4^k in which each element corresponds to a distinct k-mer. We made a vector in which the corresponding elements of the existing k-mers in the input sequence are 1. In this representation, we also included the type of RNAs. Each of the 9 possible types is shown as a one-hot vector representation. The last element of the vector is the output (or label 0 for negative interaction, 1 for positive interaction). The last 18 elements of the vector right before the output represent the type of the first and second RNAs.

Notice that this dataset does not exclude the samples in which one of the RNAs is an empty string. For such strings, the vector will be all zeros. You should exclude them when training a classifier.

Here, we use $k = 5$ to make the dataset size manageable. Therefore for each sample, we have a vector of length 1024 (first sequence) + 1024 (second sequence) + 9 (type of the first one) + 9 (type of the second one) + 1 (label).

DeepBind representation. For each character we consider a one-hot representation, i.e. a vector of size 4 in which each element corresponds to one of the characters A, C, G, or T. Each sequence of length n will be represented by a $4 \times n$ matrix in which each row corresponds to one of the 4 characters and each column to a character in the sequence. To flatten each matrix such that it fits into one line, we first write the corresponding row of A, then that of C, and then that of G, and finally that of T.

Notice that in this representation the labels are not added to the data but positive and negative data are in separate files.

Also notice that the samples which have an empty string for one of the RNAs will be shown as: e,m,p,t,y,! in the corresponding line. You should exclude them when training a classifier.

Input files. You have to train your random forest classifier off-line. The program that you submit on Canvas will contain the trained model and will be tested on the test data that we have. Your program must read from three files `seq_test.csv`, `kmer_test.csv`, and `deepbind_test.csv` in the current directory in the format explained above.

Output. Your program must output three files `seq_test.out`, `kmer_test.out`, and `deepbind_test.out` in the current directory. For each pair of RNAs in the input test file, your program must write on a line + for positive prediction and - for negative prediction.

Implementation. You can use any implementation of random forests but you are encouraged to understand the underlying algorithm so that you can tune parameters of the model to fit this particular problem.

Grading. We will test your program on one test dataset. We will measure accuracy of your model and grade using a normal curve fitted to the results of the entire class.

Upload your answer on Canvas in one zip file or tarball. Include a `README` file with running instructions and all the code/scripts you have written.