Project 4 (HMM_Dice) report

Jeet Shah

G41262537

Problem Statement: There are three three-sided dice, D1 D2 D3, with labels 1, 2, and 3. At the beginning the dealer chooses any of 3 dice with equal probability. For die Di, the probability of rolling the number j is pij. At each turn, the dealer must decide whether to: (1) keep the same die; (2) switch to one of the other two dice randomly. She chooses (1) with probability p and (2) with probability 1-p. The sequence of emissions is provided. We are required to find a sequence of states which best explains the sequence of rolls and the probability of this sequence?

Viterbi algorithm: The Viterbi algorithm helps in finding the most probable sequence of hidden states that results in a sequence of observed events. Viterbi algorithm have become standard terms for the application of dynamic programming algorithms to maximization problems involving probabilities.

There are three files. One is text file and other two are Java files. Executing “Main_fuction.java” will run this entire algorithm and give output.

Screenshots:

This is the main function. Here, the variables get loaded from input file.

Main_function.java | Viterbi_algorithm.java | InputFile.txt

```
import java.io.*;

public class Main_function {
public static void main(String[] args) throws IOException {

        float probability;
        int[] states = new int[]{0,1,2};
        int[] emissions;
        Float[][] transition_probability;
        double[][] emisssion_probability;
        double[] starting_probability = new double[]{(double) 1 / (double) 3

        File file=new File(".\\src\\InputFile.txt");
        FileReader fr=new FileReader(file);
        BufferedReader r1 = new BufferedReader(fr);

        r1.readLine();
        r1.readLine();
        r1.readLine();
        probability = Float.parseFloat(r1.readLine());
        transition_probability = new Float[][]
        {
            {probability, (1 - probability) / 2, (1 - probability) / 2},
            {(1 - probability) / 2, probability, (1 - probability) / 2},
            {(1 - probability) / 2, (1 - probability) / 2, probability},
        };

```

Here, Viterbi algorithm is implemented, which gives the sequence for given emissions.

Main_function.java | Viterbi_algorithm.java | InputFile.txt

```

public class Viterbi_algorithm {

    public  void viterbi_fun( int[] states, double[] starting_probability, Float[][] transition_probability
    {
        int[][] seq = new int[states.length][emissions.length];
        double[][] viterbi = new double[states.length][emissions.length];

        for (int s : states)   // Variables are initialized
        {
            viterbi[s][0] = starting_probability[s] * emisssion_probability[s][emissions[0] - 1];
            seq[s][0] = s;
        }

        for (int e = 1; e < emissions.length; e++) {
            int[][] temp = new int[states.length][emissions.length];

            for (int s : states) {
                double final_probability = -1;
                int current_state;
                for (int s1 : states) {
                    double current_p = viterbi[s1][e - 1] * transition_probability[s1][s] * emisssion_probal
                    if (current_p > final_probability) {
                        current_state = s1;
                        final_probability = current_p;
                        viterbi[s][e] = final_probability;  // records probability
                        System.arraycopy(seq[current_state], 0, temp[s], 0, e);  // Saves path found
                        temp[s][e] = s;
                    }
                }
            }
            seq = temp;
```

This is the input file. The input format needs to be like this.

Input file:

```
# Sample input file generated automatically at Sat Apr 11 21:23:40 EDT 2020
# All lines beginning with # are comments and can be ignored
# Probability of keeping the same dice
0.8
# Emission Probs
0.8 0.1 0.1
0.1 0.8 0.1
0.1 0.1 0.8
#
# Emissions
1,1,1,3,2,2,3,1,3,3,2,3,3,3,2,1,3,3,3,2,3,1,1,3,1,2,1,2,2,2,2,2,2,2,2,3,2,2,3,1,3,1,1,1,1,2,2
```

Output:

```
Probability of the sequence is: 1.7458098590716205E-48
Sequence is: D1 D1 D1 D3 D3 D3 D3 D3 D3 D3 D3 D3 D3 D3 D3 D3 D3 D3 D3 D1 D1 D1 D1 D1 D2 D2 D2 D2 D2 D2 D2 D2 D2 D1 D1 D1 D1 D1 D1 D2 D2 D
```