Machine Learning & Data Mining
Homework 5

Caltech CS/CNS/EE 155
Released February 9th, 2017

This set is due 9pm, February 17th, via Moodle. You are free to collaborate on all of the problems, subject to the collaboration policy stated in the syllabus.

1 Naive Bayes

TA: Emily Mazo

Question A [10 points]: The Naive Bayes classifier selects the most likely classification V_{nb} given the attributes $a_1, a_2, \ldots, a_n$. This results in

$$V_{nb} = \arg\max_{v_j \in V} P(v_j) \prod P(a_i|v_j) \tag{1}$$

We generally estimate $P(a_i|v_j)$ using to so–called "m-estimate":

$$P(a_i|v_j) = \frac{n_c + mp}{n + m}$$

where

$$\begin{aligned} n \in \mathbb{N} &= \text{the number of training examples for which } v = v_j \\ n_c \in \mathbb{N} &= \text{number of examples for which } v = v_j \text{ and } a = a_i \\ p \in [0,1] &= \text{our prior estimate for } P(a_i|v_j) \text{ (a modeling choice)} \\ m \in \mathbb{R} &= \text{our regularization parameter a.k.a. our confidence in our prior } p \text{ (a modeling choice)} \end{aligned}$$

In this problem, let $m = 3$ and assume the uniform prior: $p = 1/(\text{number-of-attribute-values}) = \frac{1}{2}$ for all our attributes.

Consider the following data set, where Color, Type, and Origin are features, and Stolen? is either yes or no:

Example No.	Color	Type	Origin	Stolen?
1	Red	Sports	Domestic	Yes
2	Red	Sports	Domestic	No
3	Red	Sports	Domestic	Yes
4	Yellow	Sports	Domestic	No
5	Yellow	Sports	Imported	No
6	Yellow	SUV	Imported	No
7	Yellow	SUV	Imported	No
8	Yellow	SUV	Domestic	No
9	Red	SUV	Imported	No
10	Red	Sports	Imported	Yes

Suppose you have a Red Domestic SUV. Given this dataset, calculate $P(\text{stolen}) \prod P(a_i|v_j)$ and $P(\text{not stolen}) \prod P(a_i|v_j)$ and the resulting V_{nb}. Would you classify your car as more likely to be stolen or not to be stolen?

Question B [5 points]: Suppose you are given a data set of the number of cups of coffee drunk by a Caltech student each day of a week, as well as the number of hours of sleep that the student had the previous night (for example, on Tuesday they drank four cups of coffee, and had five hours of sleep the night before). Note that the amount of sleep the student got on Monday would affect the number of hours they slept on Tuesday.

You would like to determine if you can predict how many cups of coffee this student will drink on any given day, based on how much sleep they had done the night before. If we were to use Naive Bayes with this data, would it work well? If not, why? Hint: think about the assumptions Naive Bayes makes.

2 Sequence Prediction

TAs: Avishek Dutta, Andrew Kang

In this problem we will explore some of the various algorithms associated with Hidden Markov Models (HMM), as discussed in lecture.

Complexity

Suppose we have a hidden Markov model (HMM), and want to determine the highest-probability state sequence given an observation. Two ways to solve this problem are the naive algorithm and the Viterbi algorithm. The naive algorithm computes the probability of each possible state sequence and returns the sequence with the highest probability.

Question A [5 points]: What is the time complexity (big-O) of the naive algorithm?

Question B [5 points]: What is the time complexity (big-O) of the Viterbi algorithm?

Concepts

Question C [5 points]: When the number of hidden states is unknown while training an HMM for a fixed observation set, if we want to increase the training data likelihood, we can do so by allowing more hidden states. True or false? Give an explanation.

Question D [5 points]: Prove that if a coefficient of the initial state or state transition probability matrices of an HMM is initially 0, it will remain 0 until the end of the EM algorithm.

Sequence Prediction

These next few problems will require extensive coding so be sure to start early. We've provided you with skeleton code for your implementations. Using this code is optional, but highly recommended as it takes care of most of the data parsing and other miscellaneous setup work for you. It will also make it easier for us to help you debug your code.

We provide you with eight different files. Six of the files are 2E.py, 2Fi.py, 2Fii.py, 2G.py, 2H.py, and 2J.py. These are scripts that can be used for running your implementations for each of the corresponding problems. The scripts provide useful output in an easy-to-read format. There is no need to modify these files. You will write your code in HMM.py, within the appropriate functions where indicated. There should be no need to write additional functions, but feel free to do so if you would like. Lastly, Utility.py contains some functions used for loading data. You will not have to modify this file.

The supplementary data folder contains 6 files titled sequence_data0.txt, sequence_data1.txt, ... , sequence_data5.txt. Each file specifies a **trained** HMM. The first row contains two tab-delimited numbers: the number of states Y and the number of types of observations X. The X observations emit outputs $0, 1, ..., X - 1$. The next Y rows of Y tab-delimited floating-point numbers describe the state transition matrix. Each row represents the current state, each column represents a state to transition to, and each entry represents the probability of that transition occurring. The next Y rows of X tab-delimited floating-point numbers describe the output emission matrix, encoded analogously to the state transition matrix. The file ends with 5 possible emissions from that HMM.

The supplementary data folder also contains one additional files titled ron.txt. This is used in problems 2G and 2H and is explained in greater detail there.

Question E [10 points]: For each of the six trained HMMs, find the max-probability state sequence for each of the five input sequences at the end of the corresponding file. To complete this problem, you will have to implement the Viterbi algorithm. Write your implementation well, as we will be reusing it in a later question.

Show your results on the 6 files. (Copy-pasting the results of 2E.py suffices.)

Question F [20 points]: For each of the six trained HMMs, find the probabilities of emitting the five input sequences at the end of the corresponding file. To complete this problem, you will have to implement the Forward algorithm and the Backward algorithm. You may assume that the initial state is randomly selected along a uniform distribution. Again, write your implementation well, as we will be reusing it in a later question.

Note that the probability of emitting an input sequence can be found by using either the α vectors from the Forward algorithm or the β vectors from the Backward algorithm. You don't need to worry about this, as it is done for you in probability_alphas() and probability_betas().

i. [10 points]: Implement the Forward algorithm. Show your results on the 6 files.

ii. [10 points]: Implement the Backward algorithm. Show your results on the 6 files.

After you complete questions 2E and 2F, you can compare your results for the file titled sequence_data0.txt with the values given in the table below:

Dataset	Emission Sequence	Max-probability State Sequence	Probability of Sequence
0	25421	31033	4.537e-05
0	01232367534	22222100310	1.620e-11
0	5452674261527433	1031003103222222	4.348e-15
0	7226213164512267255	1310331000033100310	4.739e-18
0	0247120602352051010255241	2222222222222222222222103	9.365e-24

HMM Training

Ron is an avid music listener, and his genre preferences at any given time depend on his mood. Ron's possible moods are happy, mellow, sad, and angry. Ron experiences one mood per day (as humans are known to do) and chooses one of ten genres of music to listen to that day depending on his mood. Ron's roommate, who is known to take to odd hobbies, is interested in how Ron's mood affects his music selection, and thus collects data on Rons mood and music selection for six years (2190 data points). This data is contained in the supplementary file ron.txt. Each row contains two tab-delimited strings: Ron's mood and Ron's genre preference that day. The data is split into 12 sequences, each corresponding to half a year's worth of observations. The sequences are separated by a row containing only the character '-'.

Question G [10 points]: Use a single M-step to train a supervised Hidden Markov Model on the data in ron.txt. What are the learned state transition and output emission matrices?

Question H [15 points]: Now suppose that Ron has a third roommate who is also interested in how Ron's mood affects his music selection. This roommate is lazier than the other one, so he simply steals the first roommate's data. Unfortunately, he only manages to grab half the data, namely, Ron's choice of music for each of the 2190 days.

In this question, we will train an unsupervised Hidden Markov Model on this data. Recall that unsupervised HMM training is done using the Baum-Welch algorithm and will require repeated EM steps. For this problem, use 4 hidden states and run the algorithm for 1000 iterations. The transition and observation matrices are initialized for you in the helper functions supervised_learning() and unsupervised_learning() such that they are random and normalized.

What are the learned state transition and output emission matrices?

Question I [5 points]: How do the transition and emission matrices from 2G and 2H compare? Which do you think provides a more accurate representation of Ron's moods and how they affect his music choices? Justify your answer. Suggest one way that we may be able to improve the method (supervised or unsupervised) that you believe produces the less accurate representation.

Sequence Generation

Hidden Markov Models fall under the umbrella of generative models and therefore can be used to not only predict sequential data, but also generate it.

Question J [5 points]: Load the trained HMM from the files titled sequence_data0.txt, . . . , sequence_-data5.txt. Use the six models to probabilistically generate five sequences of emissions from each model, each of length 20. Show your results. Which generated emission sequence is your favorite? Why?