Midterm practice questions

UMass CS 585 — Oct 16, 2016

1 Topics on the midterm

Language concepts

- Parts of speech
- Regular expressions, text normalization

Probability / machine learning

- Probability theory: Marginal probs, conditional probs, law(s) of total probability, Bayes Rule.
- Maximum likelihood estimation
- Naive Bayes
- Relative frequency estimation and pseudocount smoothing
- Logistic regression (for binary classification)
- Markov / N-Gram language models

Structured models

- Hidden Markov models
- Viterbi algorithm
- Log-linear models and CRFs
- (Structured) Perceptron
- CFG and CKY

2 Decoding

Question 2.1. Consider the Viterbi sequence inference algorithm for a sequence length N with K possible states. (For POS tagging, it would be: there are N tokens and K parts-of-speech.) Give the following answers in terms of N and K.

(a) What's the time complexity of Viterbi?

(b) What's the space complexity of Viterbi?

(c) What's the time complexity of enumerating all possible answers?

[Solution: Time NK^2, space NK]

Question 2.2. The greedy decoding algorithm is an alternative to Viterbi. It simply makes decisions left to right, without considering future decisions. Using $A(y_{prev}, y_{cur})$ and $B_t(y_{cur})$ additive factor score notation, it creates a predicted sequence y^* as follows:

for $t = 1..T$:
  $y_t^* \leftarrow \arg\max_{k \in tagset} A(y_{t-1}^*, k) + B_t(k)$

(a) What's the time complexity of the greedy algorithm?

(b) Unlike Viterbi, the greedy algorithm does not always find the most probable solution according to the model. Why? Give an example where it might fail. Why does Viterbi get it right? **[Solution:** See `http://people.cs.umass.edu/~brenocon/inlp2015/07-hmm-notes.html`]

3 Classification

Question 3.1. Consider training and predicting with a naive Bayes classifier for two document classes, and without pseudocounts. The word "booyah" appears once for class 1, and never for class 0. When predicting on new data, if the classifier sees "booyah", what is the posterior probability of class 1?
[Solution: 1]

Question 3.2. For a probabilistic classifier for a binary classification problem, consider the prediction rule to predict class 1 if $P(y = 1|x) > t$, and predict class 0 otherwise. This assumes some threshold t is set. If the threshold t is increased,

(a) Does precision tend to increase, decrease, or stay the same? **[Solution:** increase]

(b) Does recall tend to increase, decrease, or stay the same? **[Solution:** decrease]

4 Classifiers

Here's a naive Bayes model with the following conditional probability table (each row is that class's unigram language model):
and the following prior probabilities over classes:

word type	a	b	c
$P(w \mid y = 1)$	$5/10$	$3/10$	$2/10$
$P(w \mid y = 0)$	$2/10$	$2/10$	$6/10$

$P(y = 1)$	$P(y = 0)$
$8/10$	$2/10$

Naive Bayes

Consider a binary classification problem, for whether a document is about the end of the world (class $y = 1$), or it is not about the end of the world (class $y = 0$).

Question 4.1. Consider a document consisting of 2 `a`'s, and 1 `c`.

Note: In this practice and on the midterm, you do not need to convert to decimal or simplify fractions. You may find it easier to not simplify the fractions. On the midterm, we will not penalize simple arithmetic errors. Please show your work.

(a) What is the probability that it is about the end of the world?

(b) What is the probability it is not about the end of the world?

Question 4.2. Now suppose that we know the document is about the end of the world ($y = 1$).

(a) True or False, the naive Bayes model is able to tell us the probability of seeing the document $\vec{w} = (a, a, b, c)$ under the model.

(b) If True, what is the probability?

5 Language Models

We consider a language over the three symbols 'A', 'B', and 'C'.

Question 5.1. Consider the training 'corpus'

$$(A, C, C, B, A, B, C)$$

(a) Under a bigram language model with zero pseudocounts, what is the probability of the observation (A, B, B)? Please include generation of the END event.

(b) Under a bigram language model with a pseudocount of $\alpha = 1$, what is the probability of the observation (A, B, B)? Please include generation of the END event.

6 HMMs

Code-switching is when people switch between languages when communicating. For example, the phrase *pie a la carte* can be analyzed as code switching, where the first token *pie* is English, and the next three tokens are French.

We'll model code-switching with an HMM. The model is: at every token position t, the variable y_t denotes which language the speaker is using. y_t can be one of two states, either E or F. The word is then produced by a unigram language model for that language (this the HMM's emission distribution). Assume we know the language model parameters (i.e. the probability of a given word, given that state=English or state=French), and we only want to learn the transition parameters (i.e. the probability of switching between English, French, the START state and the END state).

We will use the example sentence $\vec{w} = (pie,\ a,\ la,\ carte)$. The unigram model parameters are, where one row is $P_{emit}(w|E)$ and the second row is $P_{emit}(w|F)$:

	pie	a	la	carte	...
English (E)	0.01	0.2	0	0	...
French (F)	0	0.1	0.1	0.01	...

We are going to treat this emission distribution as fixed. We want to learn the transition distribution. The transition distribution describes how likely a speaker is to stay in the same language, or switch to the other language. Assume that the transition distribution is initialized to be uniform between the two states (plus a little probability for the end state):

$$\begin{array}{rlrlrl} P_{trans}(E|E) & = 0.4 & P_{trans}(F|E) & = 0.4 & P(END|E) & = 0.2 \\ P_{trans}(E|F) & = 0.4 & P_{trans}(F|F) & = 0.4 & P(END|F) & = 0.2 \\ P_{trans}(E|START) & = 0.5 & P_{trans}(F|START) & = 0.5 & & \end{array}$$

Question 6.1. Bayes Rule

Only one token has ambiguity about which language it came from, so there are only two possible $(y_1..y_4)$ sequences that have non-zero probability (remember that y_t denotes which language the speaker is using at time t). For each possible sequence $\vec{y}$, write it and its posterior probability $p(\vec{y}|\vec{w})$.

Note that many terms are shared between the unnormalized probabilities, which can be ignored when computing the posterior probabilities since they are absorbed into the normalizer.

Question 6.2. Just to learn stuff We do not cover the EM algorithm in 585, but you can find out a lot about it from many sources online.[1] How you can you use EM to learn the parameters? Note: EM will not be on the midterm. This question is just if you want to learn more about NLP.

[1] We like mathematicalmonk: `https://www.youtube.com/watch?v=AnbiNaVp3eQ`

Question 6.3. Just to learn stuff Perform the first M-step. Given the posterior expectations from the last step, estimate new values of the transition parameters. You may choose either to include the generation of an END symbol, or to not include its generation.

7 Language stuff

Question 7.1. Each of the following sentences has an incorrect part-of-speech tag. Identify which one and correct it. (If you think there are multiple incorrect tags, choose the one that is the most egregious.) We'll use a very simple tag system:

- NOUN – common noun or proper noun
- PRO – pronoun
- ADJ – adjective
- ADV – adverb
- VERB – verb, including auxiliary verbs
- PREP – preposition
- DET – determiner
- X – something else

1. Colorless/ADV green/ADJ clouds/PRO sleep/VERB furiously/ADV ./X **[Solution:** clouds/NOUN]
2. She/PRO saw/VERB herself/PRO through/PREP the/ADJ looking/ADJ glass/NOUN ./X **[Solution:** the/DET]
3. Wait/NOUN could/VERB you/PRO please/X ?/X **[Solution:** Wait/VERB]

8 Perceptron

Question 8.1. In HW4 we saw an example of when the averaged perceptron outperforms the vanilla perceptron. There is another variant of the perceptron that often outperforms the vanilla perceptron. This variant is called the **voting perceptron**. Here's how the voting perceptron works:

- initialize the weight vector
- if the voting perceptron misclassifies an example at iteration i, update the weight vector and store it as w_i.
- if it makes a correct classification at iteration i, do not update the weight vector but store w_i anyway.

- To classify an example with the voting perceptron, we classify that example with each w_i and tally up the number of votes for each class. The class with the most votes is the prediction.

Despite often achieving high accuracy, the voting perceptron is rarely used in practice. Why not? **[Solution:** The voting perceptron stores every single weight vector computed. This takes $O(T * |W|)$ space to store where T is the number of iterations we train and $|W|$ is the size of the weight vector. This can be huge for many normal problems as opposed to the averaged perceptron which only require $O(|W|)$ space to store its weight vector. Similarly, the averaged perceptron can make predictions in linear time in the size of the weight vector; the voting perceptron only makes predictions in time linear in $T * |W|$ which can be much larger.]

9 CKY

Question 9.1. Here are the rules of a context free grammar:

S $\rightarrow$ NP VP	N $\rightarrow$ teacher
S $\rightarrow$ N VP	N $\rightarrow$ strikes
NP $\rightarrow$ J N	N $\rightarrow$ kids
VP $\rightarrow$ V NP	J $\rightarrow$ teacher
VP $\rightarrow$ V N	J $\rightarrow$ idle
	V $\rightarrow$ strikes
	V $\rightarrow$ idle

Create and fill in (using dynamic programming) the CYK chart that parses the sentence "Teacher strikes idle kids." How many valid parses are there? Draw the parse tree for each one.