Homework 2 Written

Q1 Sentiment Analysis & Classification

20 Points

The problems in this section are based on the material covered in Week 4.

Q1.1 Naive Bayes

10 Points

We have a training corpus consisting of three sentences and their labels:

- The cat sat in the hat, 1
- The dog sat on the log, 1
- The fish sat in the dish, 0

Suppose we train a Naive Bayes classifier on this corpus, using maximum likelihood estimation and unigram count features without any smoothing. What are the values of the parameters $p(c)$ and $p(f|c)$ for all classes c and features f? You can simply list the parameters and their values; no need to show the arithmetic. You can skip parameters with value 0, and you can leave your answers as fractions.

$c = \{0, 1\}$

$$p(0) = \frac{1}{3}, p(1) = \frac{2}{3}$$

$f = \{the, cat, sat, in, hat, dog, on, log, fish, dish\}$

$p(the|0) = \frac{2}{6} = \frac{1}{3}, p(fish|0) = \frac{1}{6}$
$p(sat|0) = \frac{1}{6}, p(in|0) = \frac{1}{6}, p(dish|0) = \frac{1}{6}$

$p(the|1) = \frac{4}{12} = \frac{1}{3}, p(sat|1) = \frac{2}{12} = \frac{1}{6}$
$p(cat|1) = \frac{1}{12}, p(in|1) = \frac{1}{12}, p(hat|1) = \frac{1}{12}$
$p(dog|1) = \frac{1}{12}, p(on|1) = \frac{1}{12}, p(log|1) = \frac{1}{12}$

What class would our Naive Bayes classifier predict for the test sentence The cat sat"? Show your work, ie. show the calculations for the predicted probabilities of both classes.

$\hat{c} = \text{argmax}_c \; p(f_1, f_2, ..., f_n|c)p(c)$
$= \text{argmax}_c \; p(f_1|c)p(f_2|c)...p(f_n|c)p(c)$
$= \text{argmax}_c \; p(c)\prod_{i=1}^{n} p(f_i|c)$

$f = \{the, cat, sat\}$ and $c = \{0, 1\}$

For $c = 0$

$$p(c)\prod_{i=1}^{n} p(f_i|c) = p(0)p(the|0)p(cat|0)p(sat|0)$$
$$= \frac{1}{3} \cdot \frac{1}{3} \cdot 0 \cdot \frac{1}{6} = 0$$

For $c = 1$

$$p(c)\prod_{i=1}^{n} p(f_i|c) = p(1)p(the|1)p(cat|1)p(sat|1)$$
$$= \frac{2}{3}\cdot\frac{1}{3}\cdot\frac{1}{12}\cdot\frac{1}{6} = \frac{1}{324}$$

$\hat{c} = \text{argmax}_c \ \{0(c=0), \frac{1}{324}(c=1)\} = 1$

$$\therefore \hat{c} = 1$$

The Naive Bayes Classifier would predict class/label 1

Save Answer Last saved on **Sep 29 at 3:19 PM**

Q1.2 Logistic Regression

5 Points

The last step of the programming component asks you to get the top k most important features for your sentiment classifier. When doing this, why do we sort by absolute value? Explain why we do this rather than sorting by the raw weight values (1-2 sentences).

Logistic Regression gives a continuous set of values. The weight vector 'w' determines the importance of a feature, i.e., how much each feature contributes to the final prediction. A feature that is unimportant or gives a neutral sentiment has a low absolute value, close to zero.
A feature that strongly contributes to the prediction will have a large absolute value; this value is a large positive number if the feature contributes highly to a positive final prediction, and it is a large negative number if the feature contributes highly to a negative

final prediction.
This is why we sort the features by their absolute value of weights(stored in coef variable) as we want the most important features whether it contributes to a positive or negative prediction.

Save Answer Last saved on **Sep 29 at 3:26 PM**

Q1.3 Gradient Descent

5 Points

Suppose you are training a model using stochastic gradient descent, and you have a held-out validation set to check the performance of your model. Your loss function gives the following values on the training and validation sets as you train:

Training Steps	Training Loss	Validation Loss
100	0.9494	0.9952
200	0.8652	0.8921
300	0.7345	0.7671
400	0.6253	0.6937
500	0.5145	0.5877
600	0.4112	0.4514
700	0.3434	0.4528
800	0.2346	0.4698
900	0.1384	0.4745
1000	0.1261	0.4778

You have saved a copy of your model every 100 training steps. Which of the 10 saved models should you use for testing? Explain your answer (1-2 sentences).

The sixth model (the model at 600 training steps) must be used for testing as the model at 600 training steps has the least Validation Loss among the 10 saved models.

Training Loss continues to reduce even past 600 steps, which is bound to happen when we train more and more. But we prefer the model that performs well on the Validation set as well because the Validation Set is a more accurate representation of the Test Set. Hence we pick the model which minimizes Loss on the Validation Set.

Save Answer Last saved on **Sep 29 at 3:30 PM**

Q2 Part of Speech Tagging

25 Points

The problems in this section are based on the material covered in Week 5.

Q2.1 HMMs & the Viterbi Algorithm

10 Points

Suppose we have a training corpus consisting of two tagged sentences:

$$\begin{matrix} \text{the} & \text{can} & \text{is} & \text{in} & \text{the} & \text{drawer} \\ DT & NN & VB & PP & DT & NN \end{matrix}$$

$$\begin{matrix} \text{The} & \text{cat} & \text{can} & \text{see} & \text{the} & \text{fish} \\ DT & NN & VB & VB & DT & NN \end{matrix}$$

Suppose we train a simple HMM part-of-speech tagger on this corpus, using maximum likelihood estimation, bigram tag transition probabilities, and a single meta-tag $< s >$ (the start tag). What are the values of the parameters $p(t_i|t_{i-1})$ and $p(w_i|t_i)$ for all tags t and words w? You can simply list the parameters and their values; no need to show the arithmetic. You can skip parameters with value 0, and you can leave your answers as fractions.

$w_1^n =$ the, can, is, in, drawer, cat, see, fish
$w_1^n =$ DT, NN, VB, PP

$$p(t_i|t_{i-1}) = \frac{c(t_{i-1}, t_i)}{c(t_{i-1})}$$

$p(DT| < s >) = \frac{2}{2} = 1\,, p(NN|DT) = \frac{4}{4} = 1\,, p(VB|NN) = \frac{2}{4} = \frac{1}{2}$
$p(PP|VB) = \frac{1}{3}\,, p(DT|PP) = \frac{1}{1} = 1\,, p(VB|VB) = \frac{1}{3}\,, p(DT|VB) = \frac{1}{3}$

$$p(w_i|t_i) = \frac{c(t_i, w_i)}{c(t_i)}$$

$p(the|DT) = \frac{4}{4} = 1\,, p(can|NN) = \frac{1}{4}\,, p(is|VB) = \frac{1}{3}$
$p(in|PP) = \frac{1}{1} = 1\,, p(drawer|NN) = \frac{1}{4}\,, p(cat|NN) = \frac{1}{4}$
$p(can|VB) = \frac{1}{3}\,, p(see|VB) = \frac{1}{3}\,, p(fish|VB) = \frac{1}{4}$

What parts of speech would the trained HMM tagger in the previous problem predict for the test sentence "The fish can see the can," using Viterbi decoding? Show your work, ie. the dynamic programming table V. You can leave your answers as fractions.

t'	p(DT|t')	p(NN|t')	p(VB|t')	p(PP|t')
DT	0	1	0	0

NN	0	0	$\frac{1}{2}$	0
VB	$\frac{1}{3}$	0	$\frac{1}{3}$	$\frac{1}{3}$
PP	1	0	0	0

w_1 = the, w_2 = fish, w_3 = can, w_4 = see, w_5 = the, w_6 = can

t	DT	NN	VB	PP
$V(1,t)$ $= \pi(t)p(w_1,t)$	1 x 1 = 1 (BP=DT)	0	0	0
$V(2,t)$ = $V(1,t'$=DT$)p(t\|t'$=DT$)p(w_2,t)$	0	1 x 1 x $\frac{1}{4}$= $\frac{1}{4}$ (BP=NN)	0	0
$V(3,t)$ = $V(2,t'$=NN$)p(t\|t'$=NN$)p(w_3,t)$	0	0 x $\frac{1}{4}$= 0	$\frac{1}{4}$x$\frac{1}{2}$x$\frac{1}{3}$= $\frac{1}{24}$ (BP=VB)	0
$V(4,t)$ = $V(3,t'$=VB$)p(t\|t'$=VB$)p(w_4,t)$	0	0	$\frac{1}{24}$x$\frac{1}{3}$x$\frac{1}{3}$= $\frac{1}{216}$ (BP=VB)	0
$V(5,t)$ = $V(4,t'$=VB$)p(t\|t'$=VB$)p(w_5,t)$	$\frac{1}{216}$x$\frac{1}{3}$x 1= $\frac{1}{648}$ (BP=DT)	0	0	0
$V(6,t)$ = $V(5,t'$=DT$)p(t\|t'$=DT$)p(w_6,t)$	0	$\frac{1}{648}$x 1 x$\frac{1}{4}$= $\frac{1}{2592}$ (BP=NN)	0	0

Final Output:

$[('the', DT), ('fish', NN), ('can', VB), ('see', VB), ('the', DT), ('can', NN)]$

Save Answer Last saved on **Sep 29 at 10:39 PM**

Q2.2 The Viterbi Algorithm

5 Points

Suppose we have an HMM tagger that uses 4-gram tag transition probabilities, ie. the parameters are $p(t_i|t_{i-1}, t_{i-2}, t_{i-3})$ and $p(w_i|t_i)$. Let T be the number of tags in the tagset, and let n be the length of the input sequence to be tagged. What is the runtime, in big-O, of the vanilla Viterbi algorithm for this HMM? What is the runtime if we use beam search Viterbi with beam size k? Briefly explain your answers (a single sentence is fine).

The time complexity of the Vanilla Viterbi algorithm for this HMM is $O(nT^4)$
In worst case, the algorithm has to go through all T members of the tagset T times to create bigrams which form the contexts for the trigrams. These trigrams are formed by going through the formed bigrams T times which in turn need to be gone through T to create contexts for the 4-gram model.

With a beam size of k, only the top 'k' tags are considered in each subsequent step, and so the runtime if we use beam search Viterbi will be $O(nTk^3)$

Save Answer Last saved on **Sep 29 at 11:29 PM**

Q2.3 Tagsets

5 Points

The Penn Treebank tagset is not the only one out there; there is also the Universal Dependencies tagset, which has less than half as many tags. For example, instead of a different tag for each tense of verb, Universal Dependencies has a single tag for all verbs, regardless of their tense. What are some advantages and disadvantages of using a smaller tagset, as opposed to a larger one? Give at least one advantage and one disadvantage and briefly explain (a single sentence each is fine).

One major advantage of using a smaller tagset is the amount of time saved to test and use the model which does or uses Part-of-Speech tagging. We have seen that just for HMM Tagger with bigram tag transition probabilities the runtime is $O(nT^2)$. By halving the size of the tagset with reduce the worst case runtime to a quarter of that amount, and this reduction is much more when the taggers use a larger n-gram tag transition.

A disadvantage of using a smaller tagset as opposed to a larger one is the possible loss in accuracy of the POS Tagger. The English language could have a perfectly sensible transition from a particular type of verb to a noun, but a transition from some other type of verb to a noun or pronoun might be grammatically incorrect and practically non-existent in normal usage. But with reduced tags such as just a single tag for all verbs not accounting for cases such as this, the performance of the Tagger that uses a smaller tagset could be considerably lower.

Save Answer

Last saved on **Sep 29 at 11:06 PM**

Q2.4 MEMMs & Feature Engineering

5 Points

Another powerful feature type for part-of-speech tagging MEMMs, in addition to word and tag n-grams, are word and tag skip-grams. For example, from the sequence "The sleepy dog", we can get two bigrams, (the, sleepy) and (sleepy, dog); one trigram, (the, sleepy, dog); and one skip-gram, (the, dog). Why do we use skip-grams when we already have bigrams and trigrams? What advantages do skip-gram features offer? Give at least one advantage and briefly explain (a single sentence is fine).

A skip-gram can be useful even when the size of training data is small. For instance, in the above example, the phrase "the dog" was not shown but it was still considered by the skip-gram. While it is possible the "the dog" might have eventually turned up elsewhere in the training data, the point to consider here is that a perfectly acceptable phrase which was not in the given training data was also used to train the model. Using a skip-gram even enables the model to represent well rare or uncommon phrases considering it takes into account speech which was not present in the training data set.

Save Answer   Last saved on **Sep 29 at 11:18 PM**

Q3 Late Penalty

0 Points

This problem intentionally left blank.

Save Answer

Save All Answers   Submit & View Submission ›