Rajashekar Mudigonda
A20556922

CS 585 Spring 2024 Written Assignment #03

Due: **Saturday, February 24, 2024 at 11:59 PM CST**
Points: **30**

Objectives:

1. (20 points) Demonstrate your understanding of the Naive Bayes classifier.
2. (10 points) Demonstrate your understanding of POS Tagging with Hidden Markov Model and Viterbi algorithm.

Problem 1 [20 pts]

Your task is to **manually** develop a Naive Bayes **SPAM** classifier / model. Training and test sets are provided below. Assume that no additional pre-processing is necessary:

Training set:	
Sample:	**Label:**
get cheap pills online	SPAM
online pharmacy pills fast	SPAM
online pills fast shipment	SPAM
my pills prescription	HAM
shipment not fast	HAM
order pills online	SPAM
fast pills delivery	SPAM
this online pharmacy	HAM
take your pills	HAM
get your pills cheap	SPAM
my online order	HAM

Test set:	
Sample:	**Label:**
order pills fast	SPAM
no prescription online pharmacy	SPAM
this order was online	HAM

A) **[3 pt]** Extract vocabulary from data (you can add / remove rows as necessary):

Vocabulary:			
Word	**count**	**Word**	**count**
get	2	Not	1
cheap	2	delivery	1
pills	8	this	1
online	6	take	1
pharmacy	2	your	2
Fast	4	order	2
shipment	2		
my	2		
prescription	1		

Total unique words in vocabulary : 15
Total number of words = 37
Total words of the class SPAM = 22
Total words of the class HAM = 15

B) **[10 pt]** Construct the model(s) (**include add-1 smoothing**) / derive all parameters. Include all formulas and detailed derivation.

Model / parameters:

Prior Probabilities

P(y=SPAM) = 6/11
P(y=HAM) = 5/11

<u>Likelihood Probabilities of words for class SPAM:</u>
P(xi | class) = count(xi, SPAM)/ sum(Count(x,y=SPAM))

Using the above likelihood equation :

P(x=get|y=SPAM) =count(get, SPAM) = 2/22
P(x=cheap|y=SPAM) =count(get, SPAM) = 2/22
P(x=pills |y=SPAM) =count(get, SPAM) = 6/22
P(x=online|y=SPAM) =count(get, SPAM) = 4/22
P(x= pharmacy |y=SPAM) =count(get, SPAM) = 1/22
P(x= Fast |y=SPAM) =count(get, SPAM) = 3/22
P(x= shipment |y=SPAM) =count(get, SPAM) = 1/22
P(x= my |y=SPAM) =count(get, SPAM) = 0/22
P(x= prescription |y=SPAM) =count(get, SPAM) = 0/22
P(x= Not |y=SPAM) =count(get, SPAM) = 0/22
P(x= delivery |y=SPAM) =count(get, SPAM) = 1/22

P(x=this|y=SPAM) =count(get, SPAM) = 0/22
P(x=take|y=SPAM) =count(get, SPAM) = 0/22
P(x=your|y=SPAM) =count(get, SPAM) = 1/22
P(x=order|y=SPAM) =count(get, SPAM) = 1/22

Adding Laplace smoothing:
Count(w) + 1/22+15 = 1/37

P(x=get|y=SPAM) =count(get, SPAM) = 3/37
P(x=cheap|y=SPAM) =count(get, SPAM) = 3/37
P(x=pills |y=SPAM) =count(get, SPAM) = 7/37
P(x=online|y=SPAM) =count(get, SPAM) = 5/37
P(x= pharmacy |y=SPAM) =count(get, SPAM) = 2/37
P(x= Fast |y=SPAM) =count(get, SPAM) = 4/37
P(x= shipment |y=SPAM) =count(get, SPAM) = 2/37
P(x= my |y=SPAM) =count(get, SPAM) = 1/37
P(x= prescription |y=SPAM) =count(get, SPAM) = 1/37
P(x= Not |y=SPAM) =count(get, SPAM) = 1/37
P(x= delivery |y=SPAM) =count(get, SPAM) = 2/37
P(x=this|y=SPAM) =count(get, SPAM) = 1/37
P(x=take|y=SPAM) =count(get, SPAM) = 1/37
P(x=your|y=SPAM) =count(get, SPAM) = 2/37
P(x=order|y=SPAM) =count(get, SPAM) = 2/37

Likelihood Probabilities of words for class HAM:
P(x=get|y=HAM) =count(get, HAM) = 0/15
P(x=cheap|y=HAM) =count(get, HAM) = 0/15
P(x=pills |y= HAM) =count(get, HAM) = 2/15
P(x=online|y= HAM) =count(get, HAM) = 2/15
P(x= pharmacy |y= HAM) =count(get, HAM) = 1/15
P(x= Fast |y= HAM) =count(get, HAM) = 1/15
P(x= shipment |y= HAM) =count(get, HAM) = 1/15
P(x= my |y= HAM) =count(get, HAM) = 2/15
P(x= prescription |y= HAM) =count(get, SPAM) = 1/15
P(x= Not |y= HAM) =count(get, HAM) = 1/15
P(x= delivery |y= HAM) =count(get, SPAM) = 0/15
P(x=this|y= HAM) =count(get, SPAM) = 1/15
P(x=take|y= HAM) =count(get, SPAM) = 1/15
P(x=your|y= HAM) =count(get, SPAM) = 1/15
P(x=order|y= HAM) =count(get, SPAM) = 1/15

Adding Laplace smoothing: count(word)+1/15+15 = (c+1)/30

P(x=get|y=HAM) =count(get, HAM) = 1/30
P(x=cheap|y=HAM) =count(get, HAM) = 1/30
P(x=pills |y= HAM) =count(get, HAM) = 3/30
P(x=online|y= HAM) =count(get, HAM) = 3/30
P(x= pharmacy |y= HAM) =count(get, HAM) = 2/30
P(x= Fast |y= HAM) =count(get, HAM) = 2/30
P(x= shipment |y= HAM) =count(get, HAM) = 2/30
P(x= my |y= HAM) =count(get, HAM) = 3/30
P(x= prescription |y= HAM) =count(get, SPAM) = 2/30
P(x= Not |y= HAM) =count(get, HAM) = 2/30
P(x= delivery |y= HAM) =count(get, SPAM) = 1/30
P(x=this|y= HAM) =count(get, SPAM) = 2/30
P(x=take|y= HAM) =count(get, SPAM) = 2/30
P(x=your|y= HAM) =count(get, SPAM) = 2/30
P(x=order|y= HAM) =count(get, HAM) = 2/30

C) **[5 pt]** Test your model (**include all derivations | ignore "unknown" words if necessary**):

Testing:

Test1(t1) : order pills fast

P(y=SPAM) * P(x=order|SPAM) * P(pills | SPAM) * P(fast|SPAM)
=6/11 * 2/37 * 7/37 * 4/37 = 336/557183 = 0.00060303 = 6.0303E-4

P(y=HAM) * P(x=order|HAM) * P(pills | HAM) * P(fast|HAM)
= 5/11 * 2/30 * 3/30 * 2/30 = 60/297000 = 0.00020202 = 2.0202E-4
From above P(Y=SPAM| t1) > P(Y=HAM | t1).
So **true positive(TP)**

Test2(t2) : no prescription online pharmacy
P(y=SPAM) * P(no|SPAM) * P(prescription|SPAM) * P(online | SPAM) * p(pharmacy| SPAM)

= 6/11 * 1/37 * 5/37 * 2/37 * 3/30 =180/16715490 = 0.00001077
= 1.077E-5

P(y=HAM) * P(no|HAM) * P(prescription|HAM) * p(online | HAM) * p(pharmacy| HAM)
= 5/11 * 2/30 * 3/30 * 2/30
= 60/297000 = 0.00020202 = 2.0202E-4

P(y=HAM) > P(SPAM)
So **False Negative (FP)**

Test3(t3): this order was online
P(y=SPAM) * P(this|SPAM) * P(order|SPAM) * p(was | SPAM) * p(online| SPAM)

6/11 *1/37 * 2/37 * 5/37 = 60/557183= 0.00010768 =1.0768E-5

P(y=HAM) * P(this|HAM) * P(order|HAM) * p(was | HAM) * p(online| HAM)

= 5/11 * 2/30 * 2/30 * 3/30 = 60 / 297000 = 0.00020202 = 2.0202E-4

P(y=HAM) > P(y=SPAM)

True Negative(TN)

D) **[2 pts]** Evaluate your model (**create confusion matrix, calculate accuracy, sensitivity (recall), precision, specificity, negative predictive value, accuracy and F-Score**):

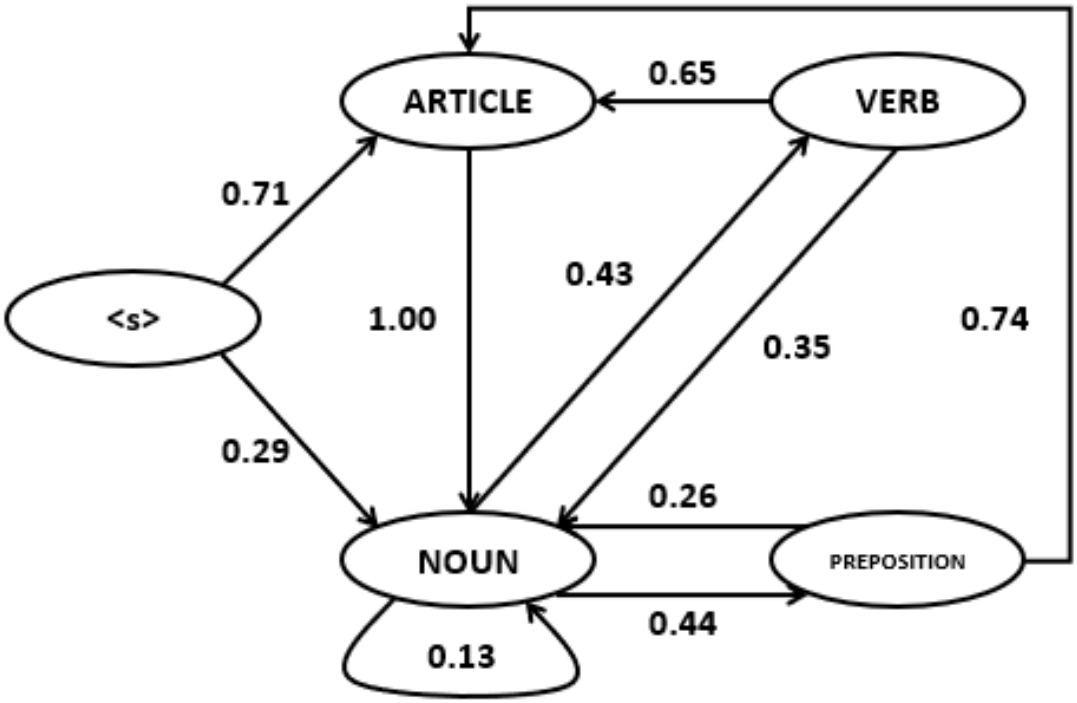
evaluation:

	Positive **SPAM**	Negative **HAM**	
Positive: **SPAM**	TP = 1(t1)	FN = 1(t2)	**Sensitivity** TP/TP+FN =1/2 = 0.5
Negative: **HAM**	0	TN = 1(t3)	**Specificity** =TN/TN+FP =1/1 + 0 = 1.0
	Precision TP /(TP + FP) = 1/1+0 = 1.0	**Negative Predictive** = TN/TN+FN = 1/1+1 = 0.5	Accuracy: 1+1/1+1+1 =2/3

F-Score: 0.5

Problem 2 [10 pts]

Given the following Hidden Markov model (transition probabilities shown; emission probabilities to be determined by you using corpus C data) based on corpus C:



And the following table of selected word counts from some corpus C:

Word/Tag	N	V	ART	P	TOTAL
flies	21	23	0	0	44
fruit	49	5	1	0	55
like	10	30	0	21	61
a	1	0	201	0	202
the	1	0	300	2	303
flower	53	15	0	0	68
flowers	42	16	0	0	58
birds	64	1	0	0	65
others	592	210	56	284	1142

TOTAL	833	300	558	307	1998

Use Viterbi algorithm to tag (with Part of Speech tags) the following sentence:

birds like the fruit

Calculating Transition Probability Matrix:

	<s>	ARTICLE	NOUN	VERB	PREPOSITION
<s>	0	0.71	0.29	0	0
ARTICLE	0	0	1.00	0	0
NOUN	0	0	0.13	0.43	0.44
VERB	0	0.65	0.35	0	0
PREPOSITION	0	0.74	0.26	0	0

Computing Observation Matrix:

	birds	like	the	fruit
<s>	0.000	0.000	0.000	0.000
ARTICLE	0.000	0.000	0.537	0.001
NOUN	0.076	0.012	0.001	0.058
VERB	0.003	0.100	0.000	0.016
PREPOSITION	0.000	0.068	0.006	0.000

Computing the Viterbi values for First Column:

V(1,) = Viterbi[s, observation]:

V(1,1) = viterbi[ARTICLE, birds] =P(ARTICLE | <s>) * P(birds | Article) = 0.71 *0 = 0.000

V(1,2) = viterbi[NOUN, birds] =P(NOUN | <s>) * P(birds | NOUN) = 0.29 *0.076 = 0.0220

V(1,3) = viterbi[VERB, birds] =P(VERB | <s>) * P(birds | VERB) = 0.0 *0.003 = 0.000

V(1,4) = viterbi[Preposition, birds] =P(Preposition | <s>) * P(birds | Preposition) = 0 *0 = 0.000

Computing the states in the second column :

V2 (1) = viterbi[ARTICLE, like] = maxs'(viterbi[state s', like] * as',1 * b1 (like)) = max(

V1 (1) * P(ARTICLE | ARTICLE) * P(like| ARTICLE) = 0.000 * 0.00 * 0.000 = 0.000

V1 (2) * P(ARTICLE | NOUN) * P(like| ARTICLE) = 0. 0220* 0.29 * 0.000 = 0.000

V1 (3) * P(ARTICLE | VERB) * P(like| ARTICLE) = 0.000 * 0.00 * 0.000 = 0.000

V1 (4) * P(ARTICLE | PREPOSITION) * P(like| ARTICLE) = 0.000 * 0.74 * 0.000= 0.000

= 0.000

V2 (2) = viterbi[NOUN, like] = maxs'(viterbi[state s', like] * as',2 * b2 (like)) = max(

V1 (1) * P(NOUN | ARTICLE) * P(like| NOUN) = 0.000 * 1.00 * 0.012 = 0.000

V1 (2) * P(NOUN | NOUN) * P(like| NOUN) = 0.0220 * 0.13 * 0.012 = 0.0001131

V1 (3) * P(NOUN | VERB) * P(like| NOUN) = 0.000 * 0.35 * 0.012 = 0.000

V1 (4) * P(NOUN | PREPOSITION) * P(like| NOUN) = 0.000 * 0.26 * 0.012 = 0.000)

V2 (2) = 0.00003432.

V2 (3) = viterbi[VERB, like] = maxs'(viterbi[state s', like] * as',3 * b3 (like)) = max(

V1 (1) * P(VERB | ARTICLE) * P(like| VERB) = 0.000 * 0.00 * 0.100 = 0.000

V1 (2) * P(VERB | NOUN) * P(like| VERB) = 0.02204 * 0.43 * 0.100 = 0.0094772

V1 (3) * P(VERB | VERB) * P(like| VERB) = 0.000 * 0.00 * 0.100 = 0.000

V1 (4) * P(VERB | PREPOSITION) * P(like| VERB) = 0.000 * 0.00 * 0.100 = 0.000)

V2 (3) = 0.0094772

3rd column:

V3 (1) = viterbi[ARTICLE, the] = maxs'(viterbi[state s', the] * as',1 * b1 (the)) = max(

V2 (1) * P(ARTICLE | ARTICLE) * P(the | ARTICLE) = 0.000 * 0.00 * 0.360 = 0.000

V2 (2) * P(ARTICLE | NOUN) * P(the | ARTICLE) = 0.0001131 * 0.00 * 0.360 = 0.000

V2 (3) * P(ARTICLE | VERB) * P(the | ARTICLE) = 0. 0094772* 0.65 * 0.537 = 0.00330802

V2 (4) * P(ARTICLE | PREPOSITION) * P(the | ARTICLE) = 0.000 * 0.74 * 0.360 = 0.000

V3 (1) = 0. 00330802

V3 (2) = viterbi[NOUN, the] = maxs'(viterbi[state s', the] * as',2 * b2 (the)) = max(

V2 (1) * P(NOUN | ARTICLE) * P(a | NOUN) = 0.000 * 1.00 * 0.001 = 0.000

V2 (2) * P(NOUN | NOUN) * P(the | NOUN) = 0. 00003432* 0.10 * 0.001 = 3.432E-9

V2 (3) * P(NOUN | VERB) * P(the | NOUN) = 0. 0094772* 0.35 * 0.001 = **3.32E-7**

V2 (4) * P(NOUN | PREPOSITION) * P(the | NOUN) = 0.000 * 0.26 * 0.001 = 0.000)

V3 (2) = **3.32E-7**

4th column:

V4 (1) = viterbi[ARTICLE, fruit] = maxs'(viterbi[state s', fruit] * as',1 * b1 (fruit)) = max(

V3 (1) * P(ARTICLE | ARTICLE) * P(fruit | ARTICLE) = 0. **00330802*** 0.00 * 0.000= 0.000

V3 (2) * P(ARTICLE | NOUN) * P(fruit | ARTICLE) = 3. 32E-7 * 0.00 * 0.000= 0.000

V3 (3) * P(ARTICLE | VERB) * P(fruit | ARTICLE) = 0.000 * 0.65 * 0.000 = 0.000

V3 (4) * P(ARTICLE | PREPOSITION) * P(fruit | ARTICLE) = 0.000 * 0.74 * 0.000= 0.000)

V4 (1)=0.000

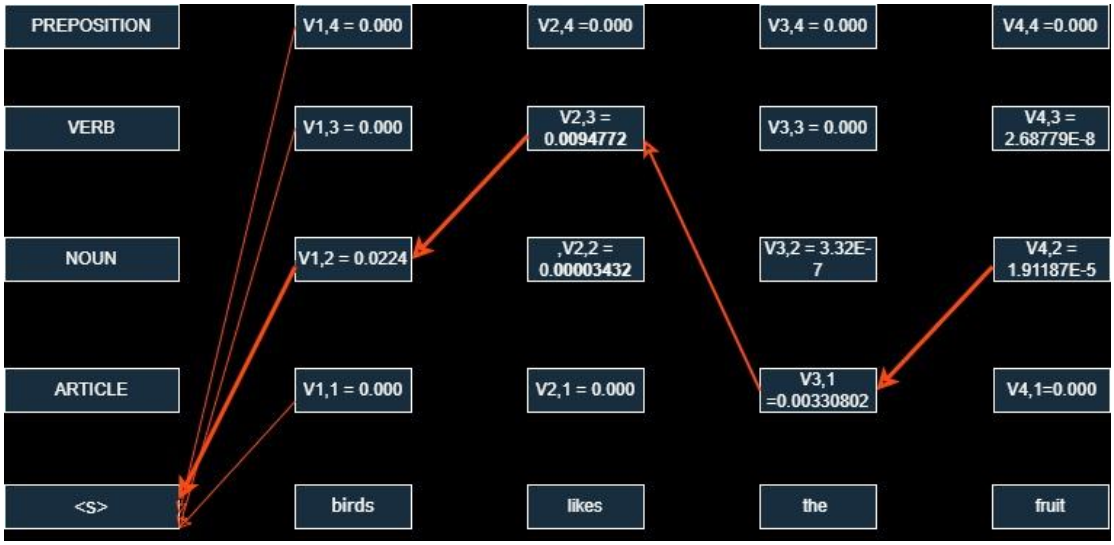
V4 (2) = viterbi[NOUN, fruit] = maxs'(viterbi[state s', fruit] * as',2 * b2 (fruit)) = max(

V3 (1) * P(NOUN | ARTICLE) * P(fruit | NOUN) = 0. 00330802* 1.00 * 0.058 = **1.91187E-5**
V3 (2) * P(NOUN | NOUN) * P(fruit | NOUN) = 3. 32E-7 * 0.10 * 0.058 = 1.98256E-9

V3 (3) * P(NOUN | VERB) * P(fuit | NOUN) = 0.000 * 0.35 * 0.058 = 0.000

V3 (4) * P(NOUN | PREPOSITION) * P(fruit | NOUN) = 0.000 * 0.26 * 0.058= 0.000

V4 (2) = **1.91187E-5**



The most probable tags are : Noun, VERB, ARTICLE, NOUN for the words in the sentence: birds, like, the, fruits