Natural Language Processing Assignment 2

Sindhuula Selvaraju(sselvar4)

February 18, 2016

Axioms:

1. p(E) = 1
2. p(X $\cup$ Y) = p(X) + p(Y) if X$\cap$Y = $\emptyset$

1 Question 1:

(a.) Given: $Y \subseteq Z$
To Prove: $p(Y) \leq p(Z)$
Proof:
Y and $(\neg Y \cap Z)$ are disjoint sets since $(\neg Y \cap Z)$ gives elements in Z not in Y.
p(Z) = p(Y $\cup$ ($\neg$Y $\cap$ Z))
p(Z) = p(Y) + p($\neg$Y $\cap$ Z) (From Axiom 2)
p(Z) $\geq$ p(Y)
$\Rightarrow p(Y) \leq$ p(Z)
Hence Proved

(b.) Given: (X| Z)
To Prove: 0$\leq$p(X| Z)$\leq$1
Proof:
We know p(X| Z) = $\frac{p(X \cap Z)}{p(Z)}$
We know that 0 $\leq$ p(Z) $\leq$ 1
We also know that in our case p(Z) $\neq$ 0 as it would make the answer not defined.
$\Rightarrow 0 < p(Z) \leq 1$
Also since X$\cap$Z $\subset$ Z hence from part (a.)
p(X$\cap$Z) $\leq$ p(Z)
$\Rightarrow$ (1.)0$\leq$ p(X cap Z) $\leq$ 1
$\Rightarrow$ (2.)p(X$\cap Z)\frac{}{p(Z) \leq}$ 1
From (1.) and (2.) we get:
$0 \leq \frac{p(X \cap Z)}{p(Z)} \leq 1$
$\Rightarrow$ 0$\leq$ p(X| Z) $\leq$ 1
Hence Proved

(c.) To Prove: p($\emptyset$) = 0
Proof:
From axiom 1: p(E) = 1
We know: (1.) E = E $\cup \emptyset$
Also: E $\cap \emptyset$ = $\emptyset$
Using axiom 2:
p(E $\cup \emptyset$) = p(E) + p($\emptyset$)
p($\emptyset$) = p(E $\cup \emptyset$) - p(E)
From (1.)
p($\emptyset$) = p(E) - p(E)
$\Rightarrow p(\emptyset) = 0$
Hence Proved

(d.) Given: $\neg$X = E - X
To Prove: p(X) = 1 - p($\neg$X)
Proof:
Since $\neg$X = E - X (Set minus)
$\Rightarrow$ (1.)$\neg$ X $\cup$ X = E
Since X and it's negation will be disjoint sets
X $\cap \neg$X = $\emptyset$
From axiom 2:
p(X $\cup \neg$ X) = p(X) + p($\neg$X)
From (1.)
p(E) = p(X) + p($\neg$X)
From axiom 1:
1 = p(X) + p($\neg$X)
p(X) = 1 - p($\neg$X)
Hence Proved

(e.) To Prove: p(singing AND rainy | rainy) = p(singing | rainy)
Let singing = X and raing = Y
To Prove: p(X AND Y | Y) = p(X | Y)
Proof:
Since X and Y are independent events:
p(X|Y) = $\frac{p(X \cap Y)}{p(y)}$
p(X AND Y | Y) = p(X $\cap$ Y | Y) = $p((X\cap$ Y)$\cap$Y)$\overline{p(Y)}$
Y $\cap$ Y = Y
p(X AND Y | Y) = p(X $\cap$ Y | Y) = $p(X\cap$ Y)$\overline{p(Y)}$ = p(X|Y)
Hence Proved

(f.) To Prove: p(X | Y) = 1 - p($\neg$X | Y)
Proof:
p(X | Y) = $\frac{p(X \cap Y)}{p(y)}$

p(¬X | Y) = $\frac{p(\neg X \cap Y)}{p(y)}$
X and ¬ X form disjoint sets.
(X ∩ Y) = Y \ (¬X ∩Y)
⇒ Y = (X∩ Y) + (¬X ∩Y)
⇒ Y = (X∩ Y) ∪ (¬X ∩Y)
p(Y) = p((X ∩ Y) ∪ (¬X ∩Y))
p(Y) = p(X ∩ Y) + p(¬X ∩ Y)
Dividing by p(Y)
1 = p(X | Y) + p(¬X | Y)
⇒ p(X| Y) = 1 - p(¬X | Y)
Hence Proved

(g.) To Simplify: (p(X|Y).p(Y)+p(X| ¬ Y).p(¬Y)).p(¬Z|X)/p(¬Z)
Proof:
p(X|Y) = $\frac{p(X \cap Y)}{p(Y)}$
p(X| ¬Y) = $\frac{p(X \cap \neg Y)}{p(\neg Y)}$
⇒ ($\frac{p(X \cap Y)}{p(Y)}$.$p(Y)$+p(X∩¬Y)$\overline{p(\neg Y)}$.p(¬Y)).p(¬Z|X)/p(¬Z)
⇒ (p(X∩ Y)+p(X∩¬ Y)).p(¬Z|X)/p(¬Z)
From axiom 2:
⇒ (p(X∩ Y) ∪ (X∩¬ Y)).p(¬Z|X)/p(¬Z)
⇒ $p(X).p$(¬Z|X)p(¬Z)
⇒p(X).p($\neg Z \cap X$)$\overline{p(X)}$ /p(¬Z)
⇒ p(¬ Z ∩ X)/p(¬Z)
⇒ p(X∩¬ Z)/p(¬Z)
⇒ p(X| ¬Z)
Hence Proved

(h.) p(singing OR rainy) = p(singing) + p(rainy)
⇒ p($singing$∪ rainy) = p(singing) + p(rainy)
This condition will only be true if the two sets are disjoint. So if I will never sing when it's rainy then this condition is true.

(i.) p(singing AND rainy) = $p(singing).p(rainy)$
⇒p(singing ∩ $rainy$) = $p(singing).p(rainy)$
⇒p(singing | rainy) = p(rainy)
⇒p(singing in the rain) = p(rainy)
This condition will only be true if I always sing when it's rainy.

(j.) Given: p(X|Y) = 0
To Prove: p(X|Y,Z) = 0
Proof:
p(X|Y) = p(X∩Y)/p(Y) = 0
⇒p(X∩Y) = 0
p(X|Y,Z) = p(X|(Y∩Z)) = p(X ∩ Y ∩ Z)/p(Y ∩ Z) = p(Z|(X∩Y)).P(X∩Y)/p(Y∩Z)

$= 0$
Hence Proved

(k.) Given: p(W|Y) = 1
To Prove: p(W|Y,Z) = 1
Proof:

$$p(W \mid Y, Z) = \frac{p(W, Y, Z)}{\sum_w p(Y, Z, W)}$$

$$p(W \mid Y, Z) = \frac{p(z \mid Y, W).p(W \mid Y).p(Y)}{\sum_w p(Z \mid Y, W).p(W \mid Y).p(Y)}$$

$$p(W \mid Y, Z) = \frac{p(Z \mid Y, W).p(W \mid Y).p(Y)}{p(Z \mid Y, W).p(W \mid Y).p(Y) + p(Z \mid Y, W).p(W \mid Y).p(Y)}$$

$$p(W \mid Y, Z) = \frac{p(Z \mid Y, W).p(W \mid Y).p(Y)}{p(Z \mid Y, W).p(W \mid Y).p(Y)}$$

$$p(W \mid Y, Z) = 1$$

2 Question 2.

(a.) p(Actual = blue) = p(Y)
p(Claimed = blue) = p(X)
p(Y | X) = $\frac{p(X|Y).p(Y)}{p(X)}$
p(Y | X) = $\frac{p(X|Y).p(Y)}{\sum_y (p(X|Y).p(Y))}$

(b.) Prior Probability = p(Actual = blue)
Likelihood of the Evidence = p(Claimed = blue | Actual = blue)
Posterior Probability = p(Actual = blue | Claimed = blue)

(c.) p(Actual = blue) = p(Y) = 0.1
p(Actual = red) = p(¬Y) = 0.9
p(Claimed = blue | Actual = blue) =p(X|Y) = 0.8
p(Claimed = blue | Actual = red) = p(X| ¬Y) = 0.2
p(Actual = blue|Claimed = blue) = p(Y| X) = $\frac{p(X|Y).p(Y)}{p(X|Y).p(Y)+p(X|\neg Y).p(\neg Y)}$
p(Actual = blue|Claimed = blue) = p(Y| X) = $\frac{(0.8)(0.1)}{(0.8)(0.1)+(0.9)(0.2)} = \frac{0.08}{0.08+0.18} =$ $\frac{0.08}{0.26} = 0.3$
Ans: p(Actual = blue|Claimed = blue) = 0.3
The judge should care about the posterior probability.

(d.) To Prove: p(A|B,Y) = $\frac{p(B|A,Y).p(A|Y)}{p(B|Y)}$
Proof:
p(A,B,Y) = p(Y,B,A)
By chain rule
p(A|B,Y).p(B|Y).p(Y)=p(Y|A,Y).p(B| A).p(A)
p(A|B,Y) = $\frac{p(Y|A,Y).p(B|A).p(A)}{p(B|Y).p(Y)}$

p(A|B,Y) = $\frac{p(Y|A,Y).p(B,A)}{p(B,Y)}$

p(A|B,Y) = $\frac{p(Y|A,Y).p(A|B).p(B)}{p(B|Y).p(B)}$

p(A|B,Y) = $\frac{p(Y|A,Y).p(A|B)}{p(B|Y)}$
Hence Proved

(e.) To Prove: p(A|B,Y) = $\frac{p(B|A,Y).p(A|Y)}{p(B|A,Y).p(A|Y)+p(B|\neg A,Y).p(\neg A|Y)}$
Proof:
By the Total Law of Probability:
p(B | Y) = $\sum_n p(B \mid Y, A_i).p(A_i \mid Y)$ = p(B | Y,A).p(A | Y)+p(B |Y,¬A).p(¬A | Y)
Using above proof with proof of part (d.)
p(A|B,Y) = $\frac{p(B|A,Y).p(A|B)}{p(B|Y)} = \frac{p(B|A,Y).p(A|B)}{p(B|Y,A).p(A|Y)+p(B|Y,\neg A).p(\neg A|Y)} = \frac{p(B|A,Y).p(A|B)}{p(B|A,Y).p(A|Y)+p(B|\neg A,Y).p(\neg A|Y)}$

(f.) p(A|B,Y) = $\frac{p(B|A,Y).p(A|B)}{p(B|A,Y).p(A|Y)+p(B|\neg A,Y).p(\neg A|Y)}$
p(A) = p(Actual = blue)
p(A|Y)=P(Actual = blue | City = Baltimore)= 0.1
p(¬ A | Y) = 0.9
p(B) = p(Claimed = blue)
p(Y) = p(City = Baltimore)
p(Claimed = blue | Actual = blue and City = Baltimore) =p(B|A,Y) = 0.8
p(Claimed = blue | Actual = red and City = Baltimore) = p(B| ¬A,Y) = 0.2
p(Actual = blue|Claimed = blue,City = Baltimore) = $\frac{p(B|A,Y).p(A|B)}{p(B|A,Y).p(A|Y)+p(B|\neg A,Y).p(\neg A|Y)}$
= $\frac{(0.8)(0.1)}{(0.8)(0.1)+(0.2)(0.9)}$

3 Question 3.

(a.)

$$\sum_{cry} p(cry \mid situation) = 1$$

(b.)

p(cry,situation)	Predator!	Timber!	I need help!	TOTAL
bwa	0	0	0.64	0.64
bwee	0	0	0.08	0.08
kiki	0.2	0	0.08	0.28
TOTAL	0.2	0	1	1

(c.)

1. This probability is written as: p(predator | kiki)
2. It can be re-written as: $\frac{p(predator \cap kiki)}{p(kiki)}$
3. This value is: $1/1.4 = 0.714$
4. Representing by Bayes Theorem = $\frac{p(kiki|predator).p(predator)}{p(kiki|predator).p(predator)+p(kiki|timber).p(timber)+p(kiki|Ineedhelp).p(Ineedhelp)}$.
5. Applying this value we get: $\frac{(1)(1)}{(1).(1)+(0.3).(1)+(0.1)(1)} = \frac{1}{1+0.3+0.1} = \frac{1}{1.4} =$ 0.714

4 Question 4.

(a.) c(BOS BOS) = Number of sentences in the corpus
c(BOS BOS i) = Number of sentences in the corpus beginning with i
c(new york EOS) = Number of sentences in corpus ending with "new york"

(b.) Sentences ending with the will very few in the trigram model because in daily use very few sentences actually end with the unless it's being used to give an example. Therefore the count of sentence in the trigram model with c(¡any word¿ the EOS) will be very low.

(c.) Expression (A) represents the probability (2)
Expression (B) represents the probability (1)
Expression (C) represents the probability (3)

5 Question 5.

We can have that the probability of a word occurring in the sentence related to the topic is known to be:
(1.)

$$p(w1) = \sum_a p(w1, a) = \sum_a p(w \mid a).p(a)$$

Applying the chain rule we will get:

p(w1w2w3w4) = p(w4 | w3,w2,w1).p(w3 | w2,w1).p(w2 | w1).p(w1)
Applying Backoff we get:
p(w1w2w3w4) = p(w4 | w3).p(w3 | w2).p(w2 | w1).p(w1)
Since w4 also depends on topic a and applying (1.):

$$p(w1w2w3w4) = \sum_{a} p(w1 \mid a).p(a).p(w2 \mid w1).p(w3 \mid w2).p(w4 \mid w3, a).p(a)$$

6 Question 8.

(a.) Words most similar to seattle:
('seahawks', 0.7584770173520642)
('spokane', 0.7537916531779193)
('tacoma', 0.7130779873872513)
('florida', 0.7101946735617494)
('atlanta', 0.6844511947308375)

Words most similar to dog:
('badger', 0.8274039535533966)
('dogs', 0.7999787247219944)
('hound', 0.7997083546063275)
('cat', 0.7923282309098834)
('borzoi', 0.7655383137965014)

Words most similar to communist:
('socialist', 0.8748215803139728)
('communists', 0.8186312375961906)
('comintern', 0.8121129342824249)
('bolshevik', 0.7949960236029026)
('leftist', 0.782475162369655)

Words most similar to jpg:
('png', 0.7579223621326954)
('svg', 0.6581031713263814)
('galleria', 0.6346978281687118)
('gif', 0.6145762699413159)
('fuji', 0.6097295357138396)

Words most similar to the:
('its', 0.7833705128890759)
('in', 0.7706653200020734)
('entire', 0.764974819915639)
('of', 0.7520807481408093)
('which', 0.7429706311725716)

Words most similar to google:

('com', 0.7459154998491606)
('yahoo', 0.7372151542300177)
('faq', 0.7262755144331383)
('flickr', 0.6973469881552221)
('web', 0.6891644937156803)

I noticed that words with commonly used alphabets and words like a,t,e,i, the etc have words with more similarity than words with rarely used characters like z,g, information etc

Smaller values of d give higher similarity values and larger values give lower similarity values both result in some new words with higher similarity than previous set of words.
Higher value of d also make the words more related to the original topic/word for example for a word like size using d=100 I got words like sizes,diameter,thickness,density,weight as the 5 most similar words which is true since they all deal with measurement.
Smaller value of d resulted in words that were comparative for example for size using d=10 I got smaller, slightly, narrower etc.

(b.) When checking for seattle seattle seattle with the new program I got the same result as the one for part (a.).
We make use of king-man to find a base to compare the analogy with.
With smaller d value get more varied results that do not go with the analogy.
With larger d values we get more accurate results.
This mainly works because it sees how similar pair of words are to each other.
NOTE: To execute run as ./findsim arg1 arg2