ANLP Tutorial Exercise Set 4 (for tutorial groups in week 8)

v1.2
School of Informatics, University of Edinburgh
Sharon Goldwater

This week's exercises focus on pointwise mutual information, logistic regression, and WSD. After completing them, you should be able to:

- Compute the estimated pointwise mutual information between two variables given appropriate data, and explain what PMI represents.
- Compute class probabilities under a given logistic regression model with weights provided.
- Describe some of the failure modes of un-regularized logistic regression models and how regularization can address them.

Exercise 1.
In lecture we introduced *pointwise mutual information* (PMI)[1] a measure of statistical independence which can tell use whether two words (or more generally, two statistical events) tend to occur together or not. We'll see it again in the Week 9 lab. The PMI between two events x and y is defined as

$$PMI(x,y) = \log_2 \frac{P(x,y)}{P(x)P(y)} \tag{1}$$

Let's consider two examples:

- x is "*eat* is the first word in a bigram" and y is "*pizza* is the second word in a bigram".
- x is "*happy* occurs in a Tweet" and y is "*pizza* occurs in a Tweet".

a) For each example, what does $P(x,y)$ represent?

b) What do negative, zero, and positive PMI values represent in terms of the statistical independence of x and y? (*Hint:* consider what must be true of the relationship between $P(x,y)$ and $P(x)P(y)$ for the PMI to be negative, zero, or positive.) Give some example pairs of words that you would expect to have negative or positive PMI (in either the bigram or Tweet scenario).

c) Normally in NLP we do not know the true probabilities of x and y so we must estimate them from data (as we will do in next week's lab with a real dataset of Tweets). Assume we use MLE to estimate probabilities. Write down an equation to compute PMI in terms of *counts* rather than *probabilities*. Use N to represent the total number of observations (e.g., the total number of bigrams in the first example, or the total number of Tweets in the second example).

d) Now, using your MLE version of PMI and the following toy dataset, compute $PMI(x,y)$, $PMI(y,z)$, and $PMI(x,z)$. You'll use the answers to error-check your code in next week's lab.

$$\begin{array}{rcrcrcl}
N & = & 12 & & & & \\
C(x) & = & 6 & \qquad & C(x,y) & = & 2 \\
C(y) & = & 4 & \qquad & C(x,z) & = & 1 \\
C(z) & = & 3 & \qquad & C(y,z) & = & 2
\end{array}$$

[1] In NLP this is sometimes also just called *mutual information*, although that term is used elsewhere for a related but not identical concept. So I'm using PMI here to avoid confusion.

Exercise 2.

Suppose we are using a logistic regression model for disambiguating three senses of the word *plant*, where y represents the latent sense.

y	sense
1	Noun: a member of the plant kingdom
2	Verb: to place in the ground
3	Noun: a factory

a) In lecture (and textbook) We saw the equation for $P(y|\vec{x})$ in a logistic regression model. Write down a simplified expression for the log probability, $\log P(y|\vec{x})$. Can you see why logistic regression models are also called *log-linear* models?

b) Imagine we have already trained the model. The following table lists the features $\vec{x}$ we are using and their weights $\vec{w}$ from training:

feat. #	feature	weight
1	`doc_contains('grow')` & y=1	2.0
2	`doc_contains('grow')` & y=2	1.8
3	`doc_contains('grow')` & y=3	0.3
4	`doc_contains('animal')` & y=1	2.0
5	`doc_contains('animal')` & y=2	0.5
6	`doc_contains('animal')` & y=3	-3.0
7	`doc_contains('industry')` & y=1	-0.1
8	`doc_contains('industry')` & y=2	1.1
9	`doc_contains('industry')` & y=3	2.7

where `doc_contains('grow')` means the document containing the target instance of *plant* also contains the word *grow*.

Now we see a new document that contains the words *industry*, *grow*, and *plant*. Compute $\sum_i w_i f_i(\vec{x}, y)$ and $P(y|\vec{x})$ for each sense y. Which sense is the most probable?

c) Now suppose we add some more features to our model:

feat. #	feature
10	`POS(tgt)=NN` & y=1
11	`POS(tgt)=NN` & y=2
12	`POS(tgt)=NN` & y=3
13	`POS(tgt)=VB` & y=1
14	`POS(tgt)=VB` & y=2
15	`POS(tgt)=VB` & y=3

where `POS(tgt)=NN` means the POS of the target word is `NN`.

We train the new model on a training set where all instances of *plant* have been annotated with sense information *and* the correct POS tag. What will happen to the weights in this model? (*Hint:* You might want to start by considering w_{14} in particular. If you can figure that one out, then start to think about the others.)

d) Suppose we stop training the new model after a large number of training iterations. We then use the model on a test set where POS tags have been added automatically (i.e., there may be errors). What problem will this cause with our WSD system? What are some ways we could change our model or training method to try to solve the problem?