CS 4700 - Foundations of Artificial Intelligence **Homework 6**
Fall 2019 **Due Tuesday, December 10th (2:40pm)**

1. **Naive Bayes for spam prediction**

Suppose you're working on 4700mail, a fancy new email provider, and you need to develop a spam filter. You have a training set of 1000 spam emails and 1000 non-spam emails to learn from. When a new email is sent, you want to predict whether it is spam or not.

For every word that's present in at least one email, we create a feature $\mathbf{x}_{i,j}$, which is 1 if email i contains word j, and 0 if email i does not contain word j. There are two classes (values of c): spam and non-spam.

Recap of Naive Bayes

In the following equations, c (the class label) is either "spam" or "non-spam."

$\mathbf{x}_{test,j}$ is 1 if word j occurs in the *test* email, and 0 if it does not occur. When you see $\mathbf{x}_{test,j}$ inside a probability, that's actually shorthand for the event that $\mathbf{x}_{test,j}$ is 1 (if the *test* email contains word j) or 0 (if the *test* email does not contain word j).

$\mathbf{x}_{test,1}, \dots, \mathbf{x}_{test,n}$ is the conjunction ("AND") of the feature values. For example, if $\mathbf{x}_{test,1} = 1, \mathbf{x}_{test,2} = 0, \mathbf{x}_{test,3} = 1, \dots$, this means "in the *test* email, word 1 occurs **and** word 2 does not occur **and** word 3 occurs, ..."

Naive Bayes predicts the class c (spam or non-spam) that maximizes the probability of the class, given the email's features (what words occur and don't occur). In other words, we predict the class c that maximizes $P(c|\mathbf{x}_{test,1}, \dots, \mathbf{x}_{test,n})$. We calculate it like this:

$$\begin{aligned}
\operatorname*{argmax}_{c\in\mathcal{C}} P(c|\mathbf{x}_{test,1}, \dots, \mathbf{x}_{test,n}) &= \operatorname*{argmax}_{c\in\mathcal{C}} \frac{P(c)\cdot P(\mathbf{x}_{test,1}, \dots, \mathbf{x}_{test,n}|c)}{P(\mathbf{x}_{test,1}, \dots, \mathbf{x}_{test,n})} && (1)\\
&= \operatorname*{argmax}_{c\in\mathcal{C}} P(c)\cdot P(\mathbf{x}_{test,1}, \dots, \mathbf{x}_{test,n}|c) && (2)\\
&\approx \operatorname*{argmax}_{c\in\mathcal{C}} P(c)\cdot \prod_{j=1}^{n} P(\mathbf{x}_{test,j}|c) && (3)
\end{aligned}$$

In line (1), we used Bayes' theorem.

In line (2), we used the fact that the denominator $P(\mathbf{x}_{test,1}, \dots, \mathbf{x}_{test,n})$ is the same no matter what c is, so taking it out doesn't change which c maximizes the expression.

In line (3), we used our **conditional independence assumption** (we assume that all of the words in the email are conditionally independent of each other, given the email's class - spam or non-spam).

Our training dataset has the following statistics. (For this problem, pretend that there are no other words in the dataset.)

Word	**# *spam* emails with the word**	**# *non-spam* emails with the word**
free	400	200
hello	550	900
probability	0	20
viagra	300	10

For parts (a) to (e), estimate probabilities using the fraction of emails (no smoothing). For example:

$$P(\mathbf{x}_{test,j} = 1|spam) \leftarrow \frac{\text{(Number of } \textbf{spam} \text{ training emails } \textbf{with word j}\text{)}}{\text{(Number of } \textbf{spam} \text{ training emails)}}$$

$$P(\mathbf{x}_{test,j} = 0|spam) \leftarrow \frac{\text{(Number of } \textbf{spam} \text{ training emails } \textbf{without word j}\text{)}}{\text{(Number of } \textbf{spam} \text{ training emails)}}$$

For all parts, you can estimate $P(c)$ in the obvious way. For example,

$$P(\text{spam}) \leftarrow \frac{\text{(Number of } \textbf{spam} \text{ training emails)}}{\text{(}\textbf{Total} \text{ number of training emails)}}$$

You do not need to use logs for this problem.

(a) In the boxed recap, when going from lines (2) to (3), Naive Bayes makes a **conditional independence** assumption to simplify $P(\mathbf{x}_{test,1}, \ldots, \mathbf{x}_{test,n}|c)$ to $\prod_{j=1}^{n} P(\mathbf{x}_{test,j}|c)$. (Specifically, each word is assumed to be conditionally independent of every other word, given the email's class – spam or non-spam.) In addition to the statistics in the table, suppose we also know that 250 **spam** training emails contain **both** the words "viagra" and "free". Is this conditional independence assumption likely to be true? Provide a brief mathematical justification.

Recall: events A and B are conditionally independent given C, iff $P(A \cap B|C) = P(A|C) \cdot P(B|C)$.

(b) Suppose there are n distinct words in our dataset. How many parameters does Naive Bayes require you to estimate? Your answer will be a formula in terms of n. Show your calculations. (Here, a parameter is any probability that needs to be estimated from the training set. Do not include probabilities that can be fully determined using 1 minus other probabilities that we've already estimated.)

(c) Suppose we think that Naive Bayes' conditional independence assumption is wrong, and we try to use line (2) in the boxed recap directly. This requires estimating the joint probability $P(\mathbf{x}_{test,1}, \ldots, \mathbf{x}_{test,n}|c)$ for all possible combinations of $\mathbf{x}_{test,1}, \ldots, \mathbf{x}_{test,n}$ and for all classes c. How many parameters would that require you to estimate? (Again, your answer will be a formula in terms of n. Again, do not include probabilities that can be fully determined using 1 minus other probabilities that we've already estimated.) Show your calculations.

(d) Compare your answers for part (b) and (c). Using these results, explain why Naive Bayes chooses to make this conditional independence assumption even when it's often violated in the real world.

(e) Now we get a new email $\mathbf{x}_{test}$ with the following text:

"probability free viagra"

Using the Naive Bayes classifier, predict whether it is spam or non-spam. Again, calculate all probabilities $P(\mathbf{x}_{test,i}|c)$ using the fraction of emails, **without smoothing**. Show your calculations.

(f) Repeat part (e) but using **additive (Laplace) smoothing with** $\alpha = 1$. For example, estimate

$$P(\mathbf{x}_{test,j} = 1|spam) \leftarrow \frac{\text{(Number of } \textbf{spam} \text{ training emails } \textbf{with word j}\text{)} + 1}{\text{(Number of } \textbf{spam} \text{ training emails)} + 2}$$

$$P(\mathbf{x}_{test,j} = 0|spam) \leftarrow \frac{\text{(Number of } \textbf{spam} \text{ training emails } \textbf{without word j}\text{)} + 1}{\text{(Number of } \textbf{spam} \text{ training emails)} + 2}$$

Do not use smoothing for the overall P(spam) and P(non-spam). Show your calculations.

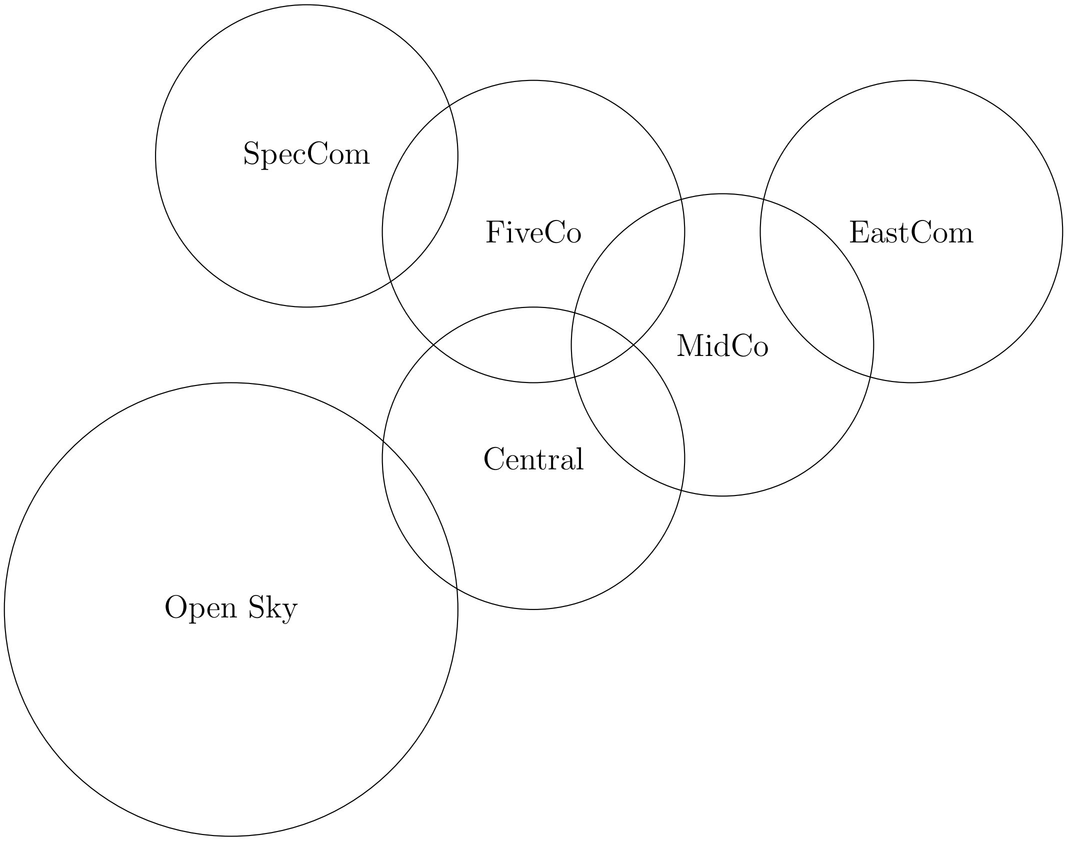
(g) In part (f), why is there a +2 in the denominator of the equations? (Your answer should include a brief description of the intuition behind smoothing.)

2. First Order & Propositional Logic

There are many different wavelengths of radio that are used for communication. For example in the United States AM radio is in the 525–1705 kHz[1] range, while FM radio is in the 87.5–108 MHz range. Wifi internet

[1] Hz is short for Hertz, which is the unit for frequency. 1 Hz means that something has a frequency of 1 / second. 1 Khz is 1,000 times/second, 1 MHz is 1,000,000/second, etc.

The Coverage Map for our Hypothetical Networks



uses the 2.4, 5, and 60 GHz frequencies. Collectively, the different wavelength ranges are reffered to as the radio spectrum. [2]

When two different broadcasters try to use the same radio frequency, they interfere with each other. That is why you cannot have two different radio stations with the same station number in the same region, or why one radio station will fade out as another fades in if you are listening to the radio while driving over long distances

Use of many parts of the spectrum are governed by international agreements, which is why wireless devices can work in multiple countries. Other parts of the spectrum are governed by national laws. Many short-range frequencies (such as 2.4 GHz) do not require specific licenses to operate, but the need to avoid collisions over longer distances means that many long range frequencies (such as the ones used for radio and local TV stations) require a license to use.

In 2015-2016 the United States and Canada decided to hold an auction to reallocate the spectrum licenses. We will not go into the details of the auction process, but a crucial detail was the ability to quickly check if a given allocation of spectrum frequencies avoided collisions.

For this exercise, we are going to consider a simplified version of this problem. Depicted is the coverage map for the different networks in our auction.

There are only 3 channels in this auction – C_1, C_2, and C_3. If two circles overlap, they cannot be assigned the same channel. For example, Open Sky and Central cannot both have channel C_1.

Predicates We define the following predicates:

[2] https://en.wikipedia.org/wiki/Broadcast_band

IsNetwork(x): True if x is a network and False otherwise
IsChannel(x): True if x is a channel and False otherwise
Overlap(x, y): True if x and y are different networks, and have overlapping coverage areas
False otherwise
HasChannel(x, a): True if x is a network and a is a channel, and if network x was given the channel a
False otherwise

Constraints We have the following constraints:

Expression	**Meaning**
$\forall x, y, c$: Overlap$(x, y) \rightarrow \neg$ (HasChannel$(x, c) \land$ HasChannel(y, c))	to be answered
to be answered	All networks have exactly one channel

At different parts of the auction process, constraints can be added to the system. In order to figure out if the problem was still solvable with the new constraints, the auction group needed to figure out if the auction was still satisfiable with the new constraints. To do this, they turned to SAT solving.

You can learn more about this auction in this lecture by Cornell alum Tim Roughgarden:
https://www.simonsfoundation.org/event/how-computer-science-informs-modern-auction-design/

(a) (Boolean values) Fill out the following table for Overlap:

Predicate	True or False?
Overlap(Open Sky, MidCo)	
Overlap(SpecCom, FiveCo)	
Overlap(EastCom, MidCo)	
Overlap(MidCo, MidCo)	
Overlap(Central, MidCo)	

(b) What does our constraint $\forall x, y, c$: Overlap$(x, y) \rightarrow \neg$ (HasChannel$(x, c) \land$ HasChannel(y, c)) mean, without using the words ”for”, ”all”, or ”implies”? (sentence)

(c) (Formula) Convert the constraint $\forall x, y, c$: Overlap$(x, y) \rightarrow \neg$ (HasChannel$(x, c) \land$ HasChannel(y, c)) into Conjunctive Normal Form.

(d) (Formula) Expressed in first order logic, what is an expression for the constraint that all networks have exactly one channel?

(e) If Open Sky and SpecCom both want channel C_1 and get it is the problem still satisfiable? If not, say why not in a sentence or short paragraph. If so, give an example of a satisfying allocation, describing what channel each station gets.

(f) If Open Sky, SpecCom, and EastCom all want channel C_1 and get it is the problem still satisfiable? If not, say why not in a sentence or short paragraph. If so, give an example of a satisfying allocation, with what channel each station gets.

3. CNF and Resolution

We will now practice using resolution in order to prove whether or not a given channel assignment can be made to work. After the warmup, this question will walk you through the resolution proof.

The first few rounds of the process have generated the constraint that Open Sky has channel C_1. We show some simple results.

Hint: This is covered on Chapter 9.5 (pages 313-317) in the text book

(a) (Warm up, formula) Give an example of a statement in propositional logic with 4 propositions, which when converted to CNF requires the propositions to be written at least 8 times. (i.e. $A \vee B \vee C \vee D$ has 4 propositions written 4 times, and $A \wedge \neg D \vee B \wedge C \vee \neg C \wedge D \vee \neg A \wedge \neg B$ has 4 propositions which are written 8 times)

(b) (Knowledge Base) First, we need to set up our knowledge base. Fill out the following table with their boolean values, assuming that Open Sky has the channel C_1. [3]

Statement	Value
HasChannel(Open Sky, C_1)	
HasChannel(Open Sky, C_2)	
HasChannel(Open Sky, C_3)	
Overlap(Open Sky, Central)	

(c) (Resolution proof) Recall your previous solution to the CNF form for the constraint $\forall x, y, c$: Overlap(x, y) $\rightarrow \neg$ (HasChannel(x, c) $\wedge$ HasChannel(y, c)). Using this and your knowledge base, prove using resolution that $\neg$HasChannel(Central, C_1) by assuming HasChannel(Central, C_1) and deriving an empty clause.

For the sake of compactness, you can abbreviate Overlap() as O(), and HasChannel() as HC().

(d) (Paragraph, or list of sentences) In plain language, what did each step of your resolution proof mean?

4. Optional: Interest only (ungraded): K Means Clustering

(a) Recall that in the k-means clustering algorithm, we first randomly initialize the cluster centers. If we run the k-means clustering algorithm two times, are we **guaranteed** to get the same clustering? If so, explain why, and if not give a simple counter-example.

(b) K-means is an algorithm that roughly tries to minimize the within-cluster sum of squares, which is a representation of variance:

$$\sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|_2^2$$

Recall that we denote the l_2 norm of a vector $x = (x_1 \ldots x_n)$ as $\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$. $S_1 \ldots S_k$ are the clusters and each μ_i is the mean of the points in S_i. Is k-means algorithm **guaranteed** to minimize this objective?

(c) Suppose now we want to minimize a different objective:

$$\sum_{i=1}^{k} \sum_{x \in S_i} \|x - \nu_i\|_1$$

Recall that we denote the l_1 norm of a vector $x = (x_1 \ldots x_n)$ as $\|x\|_1 = |x_1| + |x_2| + \ldots + |x_n|$.

i. Suppose we have a cluster S_i. What value of ν_i minimizes $\sum_{x \in S_i} \|x - \nu_i\|_1$?

ii. We can run a variation of k-means where we minimize the l_1 distance metric and consequently set the new clusterings to $\nu_1 \ldots \nu_k$ as the minimizers you derived in part i. Concretely, the new algorithm would look like this:

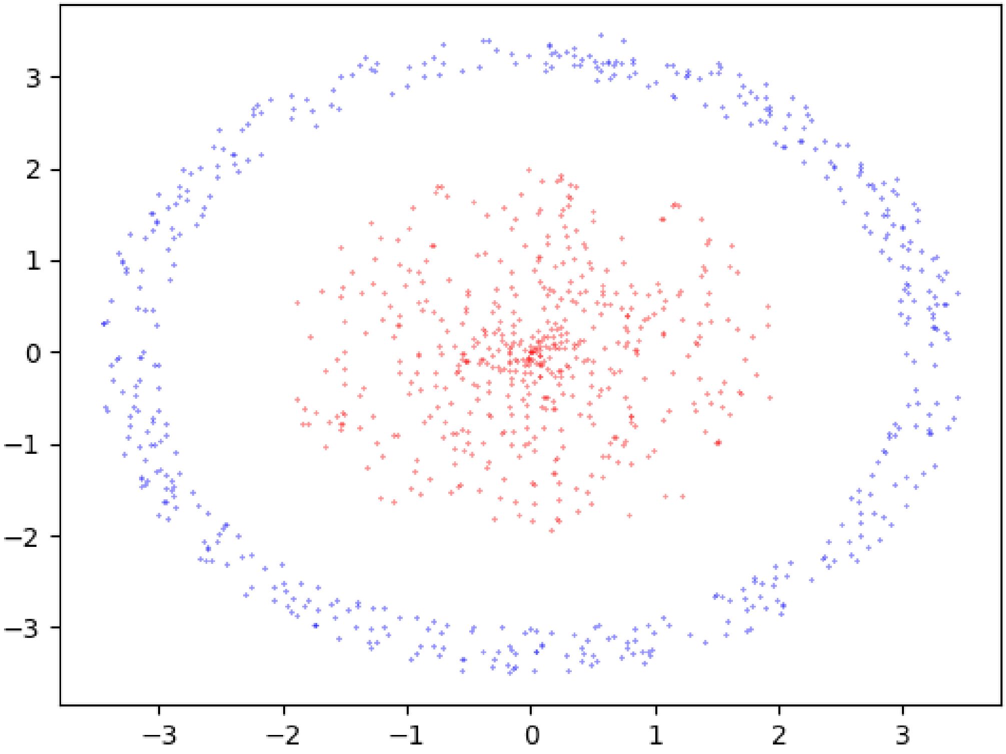
[3] Note that in the textbook we would put the statement that Overlap(Open Sky, EastCo) is false into the knowledge base by adding $\neg$Overlap(Open Sky, EastCo) to the knowledge base.

Algorithm 1 K-means with l_1 distance

```
function K-MEANS($D, k$)
    for $i = 1 \dots k$ do
        $c_i =$ random $x \in D$ not already selected
    end for
    while Stopping condition has not been reached do
        for $i = 1 \dots k$ do
            $S_i = \{x \in D \mid i = \text{argmin}_j \|c_j, x\|_1\}$
        end for
        for $i = 1 \dots k$ do
            $c_i = \nu_i$ as derived in part i
        end for
    end while
end function
```

Suppose we know for a fact that our true clusters are roughly spherical, and contain roughly the same number of points. However, due to a measurement error, one of the features on one of the data points has been replaced by a very, very large number. Which of the two algorithms (original k means or variant using l_1 norm) will achieve more accurate clusterings and why?

(d) Suppose we have a dataset that looks like the following. The red points correspond to members of one class, and the blue points correspond to members of another class.



Will the k-means algorithm be able to correctly cluster the points into the red and blue clusters? If so, explain why. If not, provide a data transformation after which k-means will be correctly cluster the points.

A data transformation is a mapping f that maps each original data point $x = (x_1, x_2)$ to a new point

$x' = (x'_1...x'_n)$

$f(x) = x'$