CSE 586 Assignment 3

Caleb Alexander

Problem 1. Implement the min-conflicts for the 8-queens problem (assume that there is exactly one queen in each column and each row). (10 points)

Answer 1. The code should be submitted alongside this pdf. A few things to note about the code,

1. The code is written in a python notebook and has been tested with python 3.12 and python 3.9 environments. It was originally written in jupyter lite, and was also tested in vscode, both worked. If it does not work on the reader's machine please reach out; so that I may investigate as to the reason.
2. The algorithm offered is certainly **not** optimized. The average loops it takes to solve a board with the algorithm is something like 380 which is not optimal.
3. It is written as a notebook because that is what I am used to, as well as a format which is quick. In no way do I guarantee that the code follows best practices for python or jupyter notebooks.

Problem 2. Explain in detail how can we construct a general and powerful spam filter using Naive Bayes Classifiers. Write your solution mathematically as discussed in class. (8 points)

Answer 2. The Naive-Bayes Classification algorithm includes 5 steps.

1. We must iterate through each word, w, that appears in our training set and calculate the probability that the word appears in and email, given that the email is spam (i.e. $P(w|S) = \frac{|spam\ emails\ containing\ w|+1}{spam\ emails}$). Notice that in the numerator we add 1 and in the denominator we add 2. This is because of *Laplace Smoothing*. Laplace smoothing is a technique used to protect against possible mis-classification due to the appearance of a word that appeared in our training set in **only** spam or **only** ham emails. It will be further elaborated later.
2. Still iterating through each word in our training set, we compute the probability of the word appearing given that the email is ham (i.e. $P(w|H) = \frac{|ham\ emails\ containing\ w|+1}{ham\ emails}$).
3. We compute the probability that any email is spam, so $P(S) = \frac{|spam\ emails|}{|all\ classified\ emails|}$.
4. We compute the probability that any email is ham, so $P(S) = \frac{|ham\ emails|}{|all\ classified\ emails|}$.
5. We are now finished training, and move on to the testing of our algorithm. Given a test email; we wish to know the probability that the email is spam given the words that appear in the email. So we:
   (a) Create a list of all the distinct words which appear in the test email, $\{x_1, \ldots, x_n\}$ (where each x_i is a distinct word). We discard any words which did not appear in training data.
   (b) Now we use the following formula,

$$P(S|x_1, \ldots, x_n) \approx \frac{P(S)\prod_{i=1}^{n} P(x_i|S)}{P(S)\prod_{i=1}^{n} P(x_i|S) + P(H)\prod_{i=1}^{n} P(x_i|H)}$$

(c) Now we establish our cut-off as 0.5. That is, if $P(S|x_1, \ldots, x_n) > 0.5$ then we classify an email as spam, otherwise we classify it as ham.

On Laplace Smoothing: Imagine that we receive a spam email which contains a word, w, that does appear in our training set but only in emails that were classified as ham. In this case, for some $x_j = w$ we have that $P(x_j|S) = 0$. Thus,

$$P(S)\prod_{i=1}^{n} P(x_i|S) = 0,$$

and $P(H)\prod_{i=1}^{n} P(x_i|H) = k$ where $k \in \mathbb{R}+$. Hence

$$P(S|x_1, \ldots, x_n) \approx \frac{P(S)\prod_{i=1}^{n} P(x_i|S)}{P(S)\prod_{i=1}^{n} P(x_i|S) + P(H)\prod_{i=1}^{n} P(x_i|H)} = \frac{0}{k} = 0.$$

So our spam email has been mis-classified as ham. A similar problem occurs in the case of receiving a ham email with a word which only appears in training spam emails. The solution to this problem is to start our count of each word at 1. This technique is called **laplace smoothing**.

Problem 3. Implement the above spam filter. (Optional group problem, 3 EXTRA credits, Due: November 10, 2024, show me your code in person)

Answer 3.

Problem 4. After your yearly checkup, the doctor has bad news and good news. The bad news is that you tested positive for a serious disease and that the test is 99% accurate. The good news is that this is a rare disease, striking only 1 in 10,000 people of your age. What are the chances that you actually have the disease? (8 points)

Answer 4. Let P^+ mean that the test is positive, and D mean that you have the disease. We know that $P(P^+|D) = 0.99$ and $P(D) = \frac{1}{10000} = 0.0001$. We wish to find $P(D|P^+)$. See that,

$$\begin{aligned} P(D|P^+) &= \frac{P(P^+|D)P(D)}{P(P^+)} = \frac{P(P^+|D)P(D)}{P(P^+|D)P(D) + P(P^+|\overline{D})P(\overline{D})} \\ &= \frac{0.99 \cdot 0.0001}{0.99 \cdot 0.0001 + 0.01 \cdot 0.9999} = \frac{0.000099}{0.010098} \approx 0.0098. \end{aligned}$$

Therefore, the chance that you have the disease is about 98 in 10000, which is less than 1 in 1000.

Problem 5. Which algorithm discussed in class has been used to schedule observations for the Hubble Space Telescope, reducing the time taken to schedule a week of observations from three weeks (!) to around 10 minutes? What is the name of a well-known CSP solver we described in class? (4 points)

Answer 5. The aforementioned algorithm is the min-conflicts algorithm.

Problem 6. We wish to transmit an n-bit message to a receiving agent. The bits in the message are independently corrupted (flipped) during transmission with ϵ probability each. With an extra parity bit sent along with the original information, a message can be corrected by the receiver if at most one bit in the entire message (including the parity bit) has been corrupted. Suppose we want to ensure that the correct message is received with probability at least $1 - \delta$. What is the maximum feasible value of n? (Only CSE 586 students)

Answer 6. Since each bit is flipped independently, if we want to know the probability that exactly k bits are flipped we can represent it as a product. If $P(k, n)$ is the probability that exactly $k \leq n$ of n bits is flipped then

$$P(k,n) = \epsilon^k (1-\epsilon)^{n+1-k} \binom{n+1}{k}.$$

In this case we are interested in $P(1, n)$ and $P(0, n)$, and we want $P(1, n) + P(0, n) \geq 1 - \delta$. So,

$$\begin{aligned}
\epsilon^1 (1-\epsilon)^{n+1-1} \binom{n+1}{1} &+ \epsilon^0 (1-\epsilon)^{n+1-0} \binom{n+1}{0} \\
&= (n+1)(\epsilon)(1-\epsilon)^n + (1-\epsilon)^{n+1} \\
&= (1-\epsilon)^n ((n+1)\epsilon + (1-\epsilon)) \\
&= (1-\epsilon)^n (n\epsilon + \epsilon + (1-\epsilon)) \\
&= (1-\epsilon)^n (n\epsilon + 1)
\end{aligned}$$

For values of $n = 1$ we have, $(1-\epsilon)^n(n\epsilon+1) = (1-\epsilon)(\epsilon+1) = \epsilon + 1 - \epsilon^2 - \epsilon = 1 - \epsilon^2$. But see that whether or not $1 - \epsilon^2 \geq 1 - \delta$ depends entirely on the individual values of δ and ϵ. This trend continues for more values of n. As such, I have begun to suspect that $1 - \delta$ is a typo, and is meant to be $1 - \epsilon$? In this case we have $(1-\epsilon)^n(n\epsilon+1) \geq 1 - \epsilon \iff (1-\epsilon)^{n-1}(n\epsilon+1) \geq 1$.

For $n = 1$, we have $(1-\epsilon)^{n-1}(n\epsilon+1) = (\epsilon+1) \geq 1$.
For $n = 2$, we have $(1-\epsilon)^{n-1}(n\epsilon+1) = (1-\epsilon)(2\epsilon+1) = 2\epsilon + 1 - 2\epsilon^2 - \epsilon = -2\epsilon^2 + \epsilon + 1 \geq 1 \iff -2\epsilon^2 + \epsilon \geq 0 \iff \epsilon^2 \leq \frac{\epsilon}{2}$. Notice that this inequality depends on ϵ, if $\epsilon \leq 0.5$ then the inequality is true, otherwise it is false.
As n increases this should continue, in that the maximum feasible n depends on ϵ. As such, I cannot find a specific integer value of n which is the maximum feasible value without knowing anything about ϵ and/or δ (if it is meant to be δ). Perhaps, I have made some large mistake, and if I have; please let me know of it while grading. I would appreciate the information.