COMS 4771 HW1 (Spring 2023)

Due: Fri Feb 17, 2023 at 11:59pm

This homework is to be done **alone**. No late homeworks are allowed. To receive credit, a typesetted copy of the homework pdf must be uploaded to Gradescope by the due date. You must show your work to receive full credit. Discussing possible approaches for solutions for homework questions is encouraged on the course discussion board and with your peers, but you must write your own individual solutions and **not** share your written work/code. You must cite all resources (including online material, books, articles, help taken from/given to specific individuals, etc.) you used to complete your work.

1 Analyzing Bayes Classifier

Consider a binary classification problem where the output variable $Y \in \{0, 1\}$ is fully determined by variables A, B and C (each in $\mathbb{R}$). In particular

$$Y = \begin{cases} 1 & \text{if } A + B + C < 7 \\ 0 & \text{otherwise} \end{cases}.$$

(i) Let each A, B and C be i.i.d. exponential random variable (with mean parameter $\lambda = 1$).

(a) Suppose the variables A and B are known but C is unknown, compute $P[Y = 1|A, B]$, the optimal Bayes classifier, and the corresponding Bayes error.

(b) If only the variable A is known but the variables B and C are unknown, compute $P[Y = 1|A]$, the optimal Bayes classifier, and the corresponding Bayes error.

(c) If none of the variables A, B, C are known, what is the Bayes classifier and the corresponding Bayes error?

(ii) Assume that variables A, B and C are independent. For known A and B, show that there exists a distribution on C for which the Bayes classification error rate can be made as close to $1/2$ as desired.

2 Classification with Asymmetric Costs

As discussed in class, occasionally the standard misclassification error is not a good metric for a given task. In this question, we consider binary classification ($\mathcal{Y} = \{0, 1\}$) where the two possible types of errors are not symmetric. Specifically, we associate a cost of $p > 0$ for outputting 0 when the class is 1, and a cost of $q > 0$ for outputting 1 when the class is 0. Furthermore, we allow the classifier to output -1, indicating an overall lack of confidence as to the label of the provided

example, and incur a cost $r > 0$. Formally, given a classifier $f : \mathcal{X} \to \{-1, 0, 1\}$ and an example $x \in \mathcal{X}$ with label $y \in \{0, 1\}$ we define the loss of f with respect to example (x, y) by:

$$\ell(f(x), y) = \begin{cases} 0 & \text{if } f(x) = y \\ p & \text{if } f(x) = 0 \text{ and } y = 1 \\ q & \text{if } f(x) = 1 \text{ and } y = 0 \\ r & \text{if } f(x) = -1 \end{cases}$$

(i) Describe a real world scenario where this model of classification may be more appropriate than the standard model seen in class.

(ii) Assume that $r < \frac{pq}{p+q}$. Let f^* be defined as:

$$f^*(x) = \begin{cases} 0 & \text{if } 0 \leq \eta(x) \leq \frac{r}{p} \\ -1 & \text{if } \frac{r}{p} < \eta(x) < 1 - \frac{r}{q} \\ 1 & \text{if } 1 - \frac{r}{q} \leq \eta(x) \leq 1 \end{cases}.$$

where $\eta(x) = \Pr[Y = 1 \mid X = x]$. Show that f^* is the Bayes classifier for this model of classification. Specifically, show that for any other classifier $g : \mathcal{X} \to \{-1, 0, 1\}$ we have: $\mathbb{E}_{x,y}\, \ell(f^*(x), y) \leq \mathbb{E}_{x,y}\, \ell(g(x), y)$.

(iii) Assume that $r \geq \frac{pq}{p+q}$. Show that the Bayes classifier is given by:

$$f^*(x) = \begin{cases} 0 & \text{if } 0 \leq \eta(x) \leq \frac{q}{p+q} \\ 1 & \text{if } \frac{q}{p+q} \leq \eta(x) \leq 1 \end{cases}.$$

where $\eta(x) = \Pr[Y = 1 \mid X = x]$.

(iv) Using the previous parts, show that when $p = q$ and $r > p/2$ the Bayes classifier is the same as the one we derived in class. Explain intuitively why this makes sense.

3 Finding (local) minima of generic functions

Finding extreme values of functions in a closed form is often not possible. Here we will develop a generic algorithm to find the extremal values of a function. Consider a smooth function $f : \mathbb{R} \to \mathbb{R}$.

(i) Recall that Taylor's Remainder Theorem states:
For any $a, b \in \mathbb{R}$, exists $z \in [a, b]$, such that $f(b) = f(a) + f'(a)(b - a) + \frac{1}{2} f''(z)(b - a)^2$.

Assuming that there exists $L \geq 0$ such that for all $a, b \in \mathbb{R}$, $|f'(a) - f'(b)| \leq L^2 |a - b|$, prove the following statement:

For any $x \in \mathbb{R}$, there exists some $\eta > 0$, such that if $\bar{x} := x - \eta f'(x)$, then $f(\bar{x}) \leq f(x)$, with equality if and only if $f'(x) = 0$.

(Hint: first show that the assumption implies that f has bounded second derivative, i.e., $f''(z) \leq L^2$ (for all z); then apply the remainder theorem and analyze the difference $f(x) - f(\bar{x})$).

(ii) Part (i) gives us a generic recipe to find a new value $\bar{x}$ from an old value x such that $f(\bar{x}) \leq f(x)$. Using this result, develop an iterative algorithm to find a local minimum starting from an initial value x_0.

(iii) Use your algorithm to find the minimum of the function $f(x) := (x-3)^2 + 4e^{2x}$. You should code your algorithm in a scientific programming language like Matlab to find the solution.

You don't need to submit any code. Instead you should provide a plot of how the minimum is approximated by your implementation as a function of the number of iterations.

(iv) Why is this technique only useful for finding local minima? Suggest some possible improvements that could help find global minima.

4 Exploring the Limits Current Language Models

Recently, OpenAI launched their Chat Generative Pre-Trained Transformer (ChatGPT), a powerful language model trained on top of GPT-3. Models like ChatGPT and GPT-3 have demonstrated remarkable performance in conversation and Q&A and overall large generative modeling. Despite their impressive performance, the sentences produced by such models are not "foolproof". Here we will explore how effectively can one distinguish between human vs. AI generated written text. The classification model which you will develop can also be used to generate new sentences in a similar fashion to ChatGPT!

One of the simplest language models is N-grams. An N-gram is a sequence of N consecutive words $w_{1:N}$. The model calculates the probability of a word w_i appearing using only the $N-1$ words before it. This local independence assumption can be considered as a Markov-type property. For the bigram model, the probability of a sequence of words $w_1, w_2, \ldots, w_n$ appearing is

$$P(w_{1:n}) = \prod_{i=1}^{n} P(w_i|w_{i-1}).$$

We use maximum likelihood estimation (MLE) to determine the conditional probabilities above. Given a training corpus $\mathcal{D}$, let $C(w_{i-1}w_i)$ denote the number of times the bigram (w_{i-1}, w_i) appears in $\mathcal{D}$. Then we can approximate the conditional probability as

$$P(w_i|w_{i-1}) = \frac{P(w_{i-1}w_i)}{P(w_{i-1})} \underset{\text{approx.}}{=} \frac{C(w_{i-1}w_i)}{C(w_{i-1})}.$$

The probability that a sequence of words is from class y is

$$P(y|w_{1:n}) = \frac{P(y, w_{1:n})}{P(w_{1:n})} = \frac{P(y)P(w_{1:n}|y)}{\sum_{\tilde{y}} P(\tilde{y})P(w_{1:n}|\tilde{y})},$$

where $P(y)$ is the prior. Formally, the bigram classifier is

$$f(w_{1:n}) = \arg\max_y P(y|w_{1:n}) = \arg\max_y P(y)P(w_{1:n}|y).$$

To address *out of vocabulary* (OOV) words (i.e. N-grams in the test set but not in the training corpus), N-gram models often employ "smoothing". One common technique is Laplacian smoothing, which assumes that every N-gram appears exactly one more time than actually observed. Given a vocabulary V, the conditional probability becomes

$$P(w_i|w_{i-1}) \underset{\text{approx.}}{=} \frac{C(w_{i-1}w_i)+1}{C(w_{i-1})+|V|}.$$

Note that the denominator is increased by $|V|$ so the probabilities sum to one.

(i) Calculate the class probability $P(y|w_{1:n})$ for a trigram model with Laplacian smoothing.

Download the datafile `humvgpt.zip`[1]. This file contains around $40,000$ human written and $20,000$ ChatGPT written entries. The data is stored in separate text files named `hum.txt` and `gpt.txt` respectively.

(ii) (a) Clean the data by removing all punctuation except "`, . ? !`" and converting all words to lower case. You may also find it helpful to add special `<START>` and `<END>` tokens for calculating N-gram probabilities. Partition 90% of the data to a training set and 10% to test set.

(b) Train a bigram and trigram model by finding the N-gram frequencies in the training corpus. Calculate the percentage of bigrams/trigrams in the test set that do not appear in the training corpus (this is called the OOV rate).

(you must submit your code to get full credit)

(c) Evaluate the model on the test set and report the classification accuracy. Which model performs better and why? Your justification should consider the bigram and trigram OOV rate.

This study will also tell you how difficult or easy it is to distinguish human vs. AI generated text!

Besides classification, N-gram models may also be used for text generation. Given a sequence of $n-1$ previous tokens $w_{i-n+2:i}$, the model selects the next word with probability

$$P(w_{i+1}=w) = P(w|w_{i-n+2:i}) \underset{\text{approx.}}{=} \frac{\exp(C(w_{i-n+2:i}w)/T)}{\sum_{\tilde{w}}\exp(C(w_{i-n+2:i}\tilde{w})/T)}.$$

In the above equation, T is referred to as the temperature.

(iii) (a) Using $T=50$, generate 5 sentences of 20 words each with the human and ChatGPT bigram/trigram models. Which text corpus and N-gram model generates the best sentences? What happens when you increase/decrease the temperature?

(b) Nowadays, language models (e.g. transformers) use an *attention mechanism* to capture long-range context. Why do sentences generated from the n-gram model often fail in this regard?

[1] The data is adapted from https://huggingface.co/datasets/Hello-SimpleAI/HC3.