

# ECE368: Probabilistic Reasoning

## Lab 1: Classification with Multinomial and Gaussian Models

You can complete this lab in a group of two. Please provide the name and student number of both members.

<b>Name:</b> Bharat Bhargava	<b>Student Number:</b> 1010380892
<b>Name:</b> Daivik Dhar	<b>Student Number:</b> 1010214260

**You should hand in:** 1) A scanned .pdf version of this sheet with your answers (file size should be under 2 MB); 2) one figure for Question 1.2.(c) and two figures for Question 2.1.(c) in the .pdf format; and 3) two Python files classifier.py and ldaqda.py that contain your code. All these files should be uploaded to Quercus.

### 1 Naïve Bayes Classifier for Spam Filtering

1. (a) Write down the estimators for  $p_d$  and  $q_d$  as functions of the training data  $\{\mathbf{x}_n, y_n\}, n = 1, 2, \dots, N$  using the technique of “Laplace smoothing”. (1 pt)

**Idea:** Laplace smoothing avoids zero probabilities for unseen words by adding a count of 1 to every word observation. Let  $S = \{n : y_n = 1\}$  be the spam emails and  $H = \{n : y_n = 0\}$  be the ham emails. The estimators are:

$$\hat{p}_d = \frac{1 + \sum_{n \in S} x_{n,d}}{D + \sum_{n \in S} \sum_{j=1}^D x_{n,j}}, \quad \hat{q}_d = \frac{1 + \sum_{n \in H} x_{n,d}}{D + \sum_{n \in H} \sum_{j=1}^D x_{n,j}}$$

where  $D$  is the vocabulary size and  $x_{n,d}$  is the count of word  $d$  in email  $n$ .

- (b) Complete function learn\_distributions in python file classifier.py based on the expressions you derived in part (a). (1 pt)
2. (a) Write down the MAP rule to decide whether  $y = 1$  or  $y = 0$  based on its feature vector  $\mathbf{x}$  for a new email  $\{\mathbf{x}, y\}$ . The  $d$ -th entry of  $\mathbf{x}$  is denoted by  $x_d$ . Please incorporate  $p_d$  and  $q_d$  in your expression. Please assume that  $\pi = 0.5$ . (1 pt)

The MAP decision rule compares the posterior probabilities. We decide  $y = 1$  if  $P(y = 1|\mathbf{x}) > P(y = 0|\mathbf{x})$ . Using Bayes’ rule and the assumption that priors are equal ( $\pi = 1 - \pi = 0.5$ ), this simplifies to comparing the likelihoods:

$$P(\mathbf{x}|y = 1) \underset{y=0}{\overset{y=1}{\gtrless}} P(\mathbf{x}|y = 0)$$

Substituting the Multinomial model distributions and taking the natural logarithm (to prevent numerical underflow), we get the final decision rule:

$$\sum_{d=1}^D x_d \ln(p_d) \underset{y=0}{\overset{y=1}{\gtrless}} \sum_{d=1}^D x_d \ln(q_d)$$

**Note:** The prior terms  $\ln(\pi)$  and  $\ln(1 - \pi)$  were removed because  $\pi = 0.5$ , so  $\ln(\pi) = \ln(1 - \pi)$  cancel out.

- (b) Complete function `classify_new_email` in `classifier.py`, and test the classifier on the testing set. The number of Type 1 errors is , and the number of Type 2 errors is .
- (c) Write down the modified decision rule in the classifier such that these two types of error can be traded off. Please introduce a new parameter to achieve such a trade-off. (0.5 **pt**)

We introduce a threshold parameter  $\tau$  to bias the decision towards one class. The modified decision rule is: Decide  $y = 1$  (Spam) if:

$$\ln P(\mathbf{x}|y = 1) + \ln P(y = 1) > \ln P(\mathbf{x}|y = 0) + \ln P(y = 0) + \tau$$

Otherwise, decide  $y = 0$  (Ham).

- Increasing  $\tau$  makes it harder to classify as Spam, decreasing Type 2 errors (False Positives) but increasing Type 1 errors.
- Decreasing  $\tau$  makes it easier to classify as Spam, decreasing Type 1 errors (False Negatives) but increasing Type 2 errors.

Write your code in file `classifier.py` to implement your modified decision rule. Test it on the testing set and plot a figure to show the trade-off between Type 1 error and Type 2 error. In the figure, the  $x$ -axis should be the number of Type 1 errors and the  $y$ -axis should be the number of Type 2 errors. Plot at least 10 points corresponding to different pairs of these two types of error in your figure. The two end points of the plot should be: 1) the point with zero Type 1 error; and 2) the point with zero Type 2 error. Please save the figure with name **nbk.pdf**. (1 **pt**)

3. Why do we need Laplace smoothing? Briefly explain what would go wrong if we do use the maximum likelihood estimators in the training process. (0.5 **pt**)