ECE368: Probabilistic Reasoning

Lab 1: Classification with Binary and Gaussian Models

Name: Calvin Kwei Hoi Tan	**Student Number:**

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 $\{\mathbf{x}_n, y_n\}, n = 1, 2, \ldots, N$ using the technique of "Laplace smoothing". (1 **pt**) $k=1$, classes $=2$, $\mathbb{I}(\cdot)=$ count

$$p_d = \frac{\sum_{n=1}^{N}\left(\mathbb{I}(y_n=1)\cdot\mathbb{I}(x_{nd}=1)\right)+1}{\sum_{n=1}^{N}\left(\mathbb{I}(y_n=1)\right)+1\times 2} \qquad q_d = \frac{\sum_{n=1}^{N}\left(\mathbb{I}(y_n=0)\cdot\mathbb{I}(x_{nd}=1)\right)+1}{\sum_{n=1}^{N}\left(\mathbb{I}(y_n=0)\right)+1\times 2}$$

   (b) Complete function learn_distributions in python file classifier.py based on the expressions. (1 **pt**)

2. (a) Write down the posterior distribution $p(y|\mathbf{x})$ as a function of $\mathbf{x}$ whose d-th entry is denoted by x_d. Please incorporate parameters p_d and q_d in your expression. Assume that $\pi = 0.5$. (0.5 **pt**)

$$P(y|\vec{x}) = \frac{P(y,\vec{x})}{P(\vec{x})} \overset{(7)}{=} \frac{p(y)\cdot\prod_{d=1}^{D}\left(P(x_d|y)\right)}{P(\vec{x})} = \begin{cases} \dfrac{\frac{1}{2}\prod_{d=1}^{D}\left(p_d^{x_d}(1-p_d)^{(1-x_d)}\right)}{P(x)}, & y=1 \\ \dfrac{\frac{1}{2}\prod_{d=1}^{D}\left(q_d^{x_d}(1-q_d)^{(1-x_d)}\right)}{P(x)}, & y=0 \end{cases}$$

indep. events

(Bern, $p(y=1)=\pi=\frac{1}{2}$)

It is better to work with the log probability $\log p(y|\mathbf{x})$ to avoid numerical underflow. Write down the MAP rule to determine the label y based on feature vector $\mathbf{x}$ of a new email. (0.5 **pt**)

$$\hat{y}_{MAP} = \arg\max_y\left(\log(P(y|\vec{x}))\right)$$

where argmax$(\cdot)$ is of above, excluding denominator, logged $\Rightarrow \prod_{d=1}^{D}$ becomes $\sum_{d=1}^{D}\log$

$$= \arg\max_y\left(\log(P(y,\vec{x}))\right) = \begin{cases} 1, & \log(P(y=1,\vec{x})) \geq \log(P(y=0,\vec{x})) \\ 0, & \log(P(y=1,\vec{x})) < \log(P(y=0,\vec{x})) \end{cases}$$

   (b) Complete function classify_new_email in classifier.py, and test the classifier on the testing set. The number of Type 1 errors is 2, and the number of Type 2 errors is 17. (1.5 **pt**)

   (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**)

$$\hat{y}_{MAP} = \arg\max_y\left(\log(P(y,\vec{x}))\right) = \begin{cases} 1, & \log(P(y=1,\vec{x})) + b \geq \log(P(y=0,\vec{x})) \\ 0, & \log(P(y=0,\vec{x})) + b < \log(P(y=0,\vec{x})) \end{cases}$$

where b is a constant bias parameter

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 **nbc.pdf**. (1 **pt**)

2 Linear/Quadratic Discriminant Analysis for Height/Weight Data

1. (a) Write down the maximum likelihood estimates of the parameters $\boldsymbol{\mu}_m$, $\boldsymbol{\mu}_f$, $\boldsymbol{\Sigma}$, $\boldsymbol{\Sigma}_m$, and $\boldsymbol{\Sigma}_f$ as functions of the training data $\{\mathbf{x}_n, y_n\}, n = 1, 2, \ldots, N$. (1 **pt**)

$$\vec{\mu}_m = \frac{1}{N_m}\sum_{n=1}^{N} \vec{x}_n \cdot \mathbb{I}(y_n=1) \qquad \vec{\Sigma}_m = \frac{1}{N_m}\sum_{n=1}^{N}\left[(\vec{x}_n-\vec{\mu}_m)(\vec{x}_n-\vec{\mu}_m)^T\right]\cdot\mathbb{I}(y_n=1)$$

$$\vec{\mu}_f = \frac{1}{N_f}\sum_{n=1}^{N} \vec{x}_n \cdot \mathbb{I}(y_n=2) \qquad \vec{\Sigma}_f = \frac{1}{N_f}\sum_{n=1}^{N}\left[(\vec{x}_n-\vec{\mu}_f)(\vec{x}_n-\vec{\mu}_f)^T\right]\cdot\mathbb{I}(y_n=2)$$

where $N_m = \sum_{n=1}^{N}\mathbb{I}(y_n=1)$

$N_f = \sum_{n=1}^{N}\mathbb{I}(y_n=2)$

$$\Rightarrow \vec{\Sigma} = \frac{1}{N}\left(N_m \cdot \vec{\Sigma}_m + N_f \cdot \vec{\Sigma}_f\right)$$

(b) In the case of LDA, write down the decision boundary as a linear equation of $\mathbf{x}$ with parameters $\boldsymbol{\mu}_m$, $\boldsymbol{\mu}_f$, and $\boldsymbol{\Sigma}$. Note that we assume $\pi = 0.5$. (0.5 **pt**)

Decision Boundary:

$$\vec{x}^T\vec{\Sigma}^{-1}\vec{\mu}_m - \frac{1}{2}\vec{\mu}_m^T\vec{\Sigma}^{-1}\vec{\mu}_m + \cancel{\log(\pi)} \underset{\hat{y}=2}{\overset{\hat{y}=1}{\gtrless}} \vec{x}^T\vec{\Sigma}^{-1}\vec{\mu}_f - \frac{1}{2}\vec{\mu}_f^T\vec{\Sigma}^{-1}\vec{\mu}_f + \cancel{\log(1-\pi)}$$

$\log(\pi) = \log(1-\pi)$ since $\pi = 1/2$

In the case of QDA, write down the decision boundary as a quadratic equation of $\mathbf{x}$ with parameters $\boldsymbol{\mu}_m$, $\boldsymbol{\mu}_f$, $\boldsymbol{\Sigma}_m$, and $\boldsymbol{\Sigma}_f$. Note that we assume $\pi = 0.5$. (0.5 **pt**)

Decision Boundary:

$$-\frac{1}{2}(\vec{x}-\vec{\mu}_m)^T\vec{\Sigma}_m^{-1}(\vec{x}-\mu_m) - \log|\Sigma_m|^{1/2} + \cancel{\lg(\pi)} \underset{\hat{y}=2}{\overset{\hat{y}=1}{\gtrless}} -\frac{1}{2}(\vec{x}-\mu_f)^T\vec{\Sigma}_f^{-1}(\vec{x}-\mu_f) - \log|\vec{\Sigma}_f|^{1/2} + \cancel{\lg(1-\pi)}$$

$\det(\Sigma)$

$\lg(\pi) = \lg(1-\pi)$ since $\pi = 1/2$

(c) Complete function discrimAnalysis in ldaqda.py to visualize LDA and QDA models and the corresponding decision boundaries. Please name the figures as lda.pdf, and qda.pdf. (1 **pt**)

2. The misclassification rates are $\boxed{11.8\%}$ for LDA, and $\boxed{10.9\%}$ for QDA. (1 **pt**)