Problem set 4

- **Problem 1 (6 pts)**
In this question, you will derive the maximum likelihood estimates for the Gaussian Naïve Bayes classifier in which a random discrete class label $Y \in [K] := \{1, 2, \ldots, K\}$ and a random feature $X \in \mathbb{R}^p$ satisfy

$$\mathbb{P}(Y = k) = \pi_k, \qquad X \mid Y = k \sim N_p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \qquad \forall k \in [K]. \tag{0.1}$$

Here $\pi_1, \ldots, \pi_K$ are the priors of the class label Y, and conditioning on $Y = k$ for any $k \in [K]$, the feature vector $X \in \mathbb{R}^p$ has a p-dimensional Gaussian density with mean $\boldsymbol{\mu}_k \in \mathbb{R}^p$ and a diagonal covariance matrix

$$\boldsymbol{\mu}_k = \begin{bmatrix} \mu_{k1} \\ \vdots \\ \mu_{kp} \end{bmatrix}, \qquad \boldsymbol{\Sigma}_k = \begin{bmatrix} \sigma_{k1}^2 & 0 & \cdots & 0 \\ 0 & \sigma_{k2}^2 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \sigma_{kp}^2 \end{bmatrix}.$$

Let $(y_1, \mathbf{x}_1), \ldots, (y_n, \mathbf{x}_n)$ be n i.i.d. realizations of (Y, X).

1. **(3 pts)** Write down the log-likelihood function of $(y_1, \mathbf{x}_1), \ldots, (y_n, \mathbf{x}_n)$.
Hint: Let Z be a categorical variable taking values from $\{1, \ldots, K\}$ with corresponding probabilities $\theta_1, \ldots, \theta_K$ with $\theta_k \geq 0$ and $\sum_k \theta_k = 1$. Its probability mass function at any $Z = z$ is

$$\mathbb{P}(Z = z) = \prod_{k=1}^{K} \theta_k^{1\{z=k\}}.$$

SOLUTION: Let $\theta = (\pi_1, \ldots, \pi_K, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_K)$. Write $f_k(\mathbf{x})$ as the p.d.f. of $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ for $k \in [K]$, that is,

$$\begin{aligned} f_k(\mathbf{x}) &= \frac{1}{(2\pi)^{p/2} |\boldsymbol{\Sigma}_k|^{1/2}} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right\} \\ &= \frac{1}{(2\pi)^{p/2} (\prod_{j=1}^{p} \sigma_{kj}^2)^{1/2}} \exp\left\{ -\sum_{j=1}^{p} \frac{1}{2\sigma_{kj}^2} (x_j - \mu_{kj})^2 \right\}. \end{aligned}$$

The second line uses the diagonal structure of $\boldsymbol{\Sigma}_k$. For $(y_1, \mathbf{x}_1)$, its likelihood is

$$L(\theta) = \prod_{k=1}^{K} [\pi_k f_k(\mathbf{x}_1)]^{1\{y_i = k\}}$$

Using the previous equation, the joint log-likelihood of $(y_1, \mathbf{x}_1), \dots, (y_n, \mathbf{x}_n)$ is

$$\begin{aligned}
\ell(\theta) &= \sum_{i=1}^n \sum_{k=1}^K 1\{y_i = k\}[\log(\pi_k) + \log f_k(\mathbf{x}_i)] \\
&= \sum_{i=1}^n \sum_{k=1}^K 1\{y_i = k\}\left[\log(\pi_k) - \frac{p}{2}\log(2\pi) - \frac{1}{2}\sum_{j=1}^p \log(\sigma_{kj}^2) - \sum_{j=1}^p \frac{1}{2\sigma_{kj}^2}(x_{ij} - \mu_{kj})^2\right] \\
&= -\frac{nKp}{2}\log(2\pi) + \sum_{i=1}^n \sum_{k=1}^{K-1} 1\{y_i = k\}\left[\log(\pi_k) - \frac{1}{2}\sum_{j=1}^p \log(\sigma_{kj}^2) - \sum_{j=1}^p \frac{1}{2\sigma_{kj}^2}(x_{ij} - \mu_{kj})^2\right] \\
&\quad + \sum_{i=1}^n 1\{y_i = K\}\left[\log\left(1 - \sum_{k=1}^{K-1}\pi_k\right) - \frac{1}{2}\sum_{j=1}^p \log(\sigma_{Kj}^2) - \sum_{j=1}^p \frac{1}{2\sigma_{Kj}^2}(x_{ij} - \mu_{Kj})^2\right].
\end{aligned}$$

2. **(3 pts)** Derive the maximum likelihood estimators of π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ for all $k \in [K]$. You may assume $\sum_{i=1}^n 1\{y_i = k\} > 0$ for all $k \in [K]$.

SOLUTION: Pick any $k \in [K]$ and $j \in [p]$. Taking the partial derivative w.r.t. μ_{kj} and σ_{kj}^2 gives

$$\begin{aligned}
\frac{\partial \ell(\theta)}{\partial \mu_{kj}} &= \sum_{i=1}^n \frac{1\{y_i = k\}}{\sigma_{kj}^2}(x_{ij} - \mu_{kj}), \\
\frac{\partial \ell(\theta)}{\partial \sigma_{kj}^2} &= \sum_{i=1}^n 1\{y_i = k\}\left[-\frac{1}{2\sigma_{kj}^2} + \frac{1}{2\sigma_{kj}^4}(x_{ij} - \mu_{kj})^2\right].
\end{aligned}$$

Letting the above equal to zero and solving for $\widehat{\mu}_{kj}$ and $\widehat{\sigma}_{kj}^2$ give

$$\begin{aligned}
\widehat{\mu}_{kj} &= \frac{1}{n_k}\sum_{i=1}^n 1\{y_i = k\}x_{ij}, \qquad\qquad n_k = \sum_{i=1}^n 1\{y_i = k\}; \\
\widehat{\sigma}_{kj}^2 &= \frac{1}{n_k}\sum_{i=1}^n 1\{y_i = k\}(x_{ij} - \widehat{\mu}_{kj})^2.
\end{aligned}$$

Furthermore, taking the partial derivative w.r.t. π_a, for $1 \le a \le K-1$, gives

$$\frac{\partial \ell(\theta)}{\partial \pi_a} = \sum_{i=1}^n \frac{1\{y_i = a\}}{\pi_a} - \sum_{i=1}^n \frac{1\{y_i = K\}}{1 - \sum_{k=1}^{K-1}\pi_k} = \frac{n_a}{\pi_a} - \frac{n_K}{1 - \sum_{k=1}^{K-1}\pi_k}$$

where $n_k = \sum_{i=1}^n 1\{y_i = k\}$ for each $k \in [K]$. The MLEs of $\pi_1, \dots, \pi_{K-1}$ satisfy

$$n_a\left(1 - \sum_{k=1}^{K-1}\widehat{\pi}_k\right) = n_K\widehat{\pi}_a, \qquad \forall\ 1 \le a \le K-1. \tag{0.2}$$

Summing over $1 \le a \le K-1$ yields

$$\sum_{a=1}^{K-1} n_a\left(1 - \sum_{k=1}^{K-1}\widehat{\pi}_k\right) = n_K\sum_{a=1}^{K-1}\widehat{\pi}_a \quad \Leftrightarrow \quad (n - n_K)\left(1 - \sum_{k=1}^{K-1}\widehat{\pi}_k\right) = n_K\sum_{k=1}^{K-1}\widehat{\pi}_k.$$

Hence

$$\sum_{k=1}^{K-1} \widehat{\pi}_k = \frac{n - n_K}{n}.$$

Plugging into (0.2) further gives

$$\widehat{\pi}_a = \frac{n_a}{n}, \qquad \forall\ 1 \leq a \leq K-1,$$

completing the proof.

- **Problem 2 (14 pts)**
  For this question you will build classifiers to label images of handwritten digits. Each image is 8 by 8 pixels and is represented as a vector of dimension 64 by listing all the pixel values in raster scan order. The images are grayscale and the pixel values are between 0 and 1. The labels y are $\{0, 1, 2, \dots, 9\}$ corresponding to which character was written in the image. There are 700 training points and 400 test points for each digit; they can be found in `digits_train.txt` and `digits_test.txt`. These data sets can be loaded by using the helper function in `utils.R`. For example,

```
source("Q2_starter/utils.R")
data_train <- Load_data("Q2_starter/data/digits_train.txt")
x_train <- train$x
y_train <- train$y
```

The first 10 digits are shown in Figure 1 (the code for visualizing the dataset is located at `utils.py` in case you want to play with it).

Figure 1: The first 10 digits in the training data set

You will implement linear discriminant analysis (LDA), quadratic discriminant analysis (QDA) and the Gaussian Naïve Bayes (NB) in **Problem 1** to classify these digits. Recall that conditioning on each class $k \in \{0, 1, \dots, 9\}$, the feature $X \mid Y = k$ follows a multivariate Gaussian distribution, that is, the p.d.f. of $X = \mathbf{x} \mid Y = k$ is

$$f_k(\mathbf{x}) = (2\pi)^{-p/2} |\mathbf{\Sigma}_k|^{-1/2} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^\top \mathbf{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right\} \tag{0.3}$$

where $\boldsymbol{\mu}_k \in \mathbb{R}^p$ is the conditional mean and $\mathbf{\Sigma}_k \in \mathbb{R}^{p \times p}$ is the conditional covariance matrix. For LDA, $\mathbf{\Sigma}_k$ is assumed to be the same across classes. The priors are

$$\pi_k = \mathbb{P}(Y = k), \qquad \text{for all} \;\; k \in \{0, 1, \dots, 9\}.$$

You will compute the maximum likelihood estimators of the priors π_k, the conditional means $\boldsymbol{\mu}_k$ and the conditional covariance matrices $\mathbf{\Sigma}_k$ for $k \in \{0, 1, \dots, 9\}$, and use the estimators to construct classifiers.

Read carefully the structure of `discriminant_analysis.R`. Include your code for all subquestions.

1. **(4 pts)** Complete the functions `Comp_priors`, `Comp_cond_means` and `Comp_cond_covs` in the file `discriminant_analysis.R`.
2. **(2 pts)** Complete the functions `Predict_posterior` and `Predict_labels` in the file `discriminant_analysis.R`.
3. **(2 pts)** Use LDA to classify the test data by completing part a in `Q2_starter.R`. Report the misclassification error of LDA.
4. **(2 pts)** Use QDA to classify the test data by completing part b in `Q2_starter.R`. Report the misclassification error of QDA.
5. **(2 pts)** Use NB to classify the test data by completing part c in `Q2_starter.R`. Report the misclassification error of NB.
6. **(2 pts)** Complete part d in `Q2_starter.R`, i.e. perform LDA and QDA by using the built-in `lda` and `qda` functions and compare with your implementation in terms of both misclassification rates and computational speed.

- **Problem 3 (20 pts)**

  In this problem, you will implement logistic regression by completing the provided code in `penalized_logistic_regression.R` & `Q3_starter.R` and experiment with the completed code.

  Throughout this problem, you will be working with a subset of hand-written digits, 2's and 3's, represented as 16×16 pixel arrays. The pixel intensities are between 0 and 1, and were read into the vectors in a raster-scan manner. You are given one training set: `train` which contains 300 examples of each class. You can access and load this training set by using functions

```
source("Q3_starter/utils.R")
data_train <- Load_data("Q3_starter/data/train.csv")
x_train <- train$x
y_train <- train$y
```

  `y_train` contains the labels of these 300 images while `x_train` are the 256 pixel values of each image. You are also given a validation set that you should use for tuning and a test set that you should use for reporting the final performance.

  You need to implement the penalized logistic regression model by minimizing the cost

$$\mathcal{J}(\boldsymbol{\beta}, \beta_0) \;:=\; -\frac{1}{n}\sum_{i=1}^{n}\Big\{y_i \log\big[p(\boldsymbol{x}_i; \boldsymbol{\beta}, \beta_0)\big] + (1 - y_i)\log\big[1 - p(\boldsymbol{x}_i; \boldsymbol{\beta}, \beta_0)\big]\Big\} + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2$$

  over $(\boldsymbol{\beta}, \beta_0) \in (\mathbb{R}^p, \mathbb{R})$, where

$$p(\boldsymbol{x}_i; \boldsymbol{\beta}, \beta_0) = \frac{e^{\beta_0 + \boldsymbol{x}_i^\top \boldsymbol{\beta}}}{1 + e^{\beta_0 + \boldsymbol{x}_i^\top \boldsymbol{\beta}}}.$$

  Here n is the total number of data points, p is the number of features in $\boldsymbol{x}_i$, $\lambda \geq 0$ is the regularization parameter, and $\boldsymbol{\beta}$ and β_0 are the parameters to optimize over. Note that we should only penalize the coefficient parameters $\boldsymbol{\beta}$ and not the intercept term β_0.

  1. (**2 pts**) Prove that the gradients of $\mathcal{J}(\boldsymbol{\beta}, \beta_0)$ at any $(\bar{\boldsymbol{\beta}}, \bar{\beta}_0)$ have the following expression,

$$\begin{aligned}
\frac{\partial \mathcal{J}(\boldsymbol{\beta}, \beta_0)}{\partial \boldsymbol{\beta}}\Big|_{\bar{\boldsymbol{\beta}}, \bar{\beta}_0} &= \frac{1}{n}\sum_{i=1}^{n}\left[-y_i + \frac{e^{\bar{\beta}_0 + \boldsymbol{x}_i^\top \bar{\boldsymbol{\beta}}}}{1 + e^{\bar{\beta}_0 + \boldsymbol{x}_i^\top \bar{\boldsymbol{\beta}}}}\right]\boldsymbol{x}_i + \lambda\bar{\boldsymbol{\beta}}, \\
\frac{\partial \mathcal{J}(\boldsymbol{\beta}, \beta_0)}{\partial \beta_0}\Big|_{\bar{\boldsymbol{\beta}}, \bar{\beta}_0} &= \frac{1}{n}\sum_{i=1}^{n}\left[-y_i + \frac{e^{\bar{\beta}_0 + \boldsymbol{x}_i^\top \bar{\boldsymbol{\beta}}}}{1 + e^{\bar{\beta}_0 + \boldsymbol{x}_i^\top \bar{\boldsymbol{\beta}}}}\right].
\end{aligned}$$

  2. (**3 pts**) Implement the functions `Evaluate`, `Predict_logis`, `Comp_gradient` and `Comp_loss` located at `penalized_logistic_regression.R`. While implementing the functions, remember to vectorize the operations (use vector / matrix operations such as addition and different multiplications); you should not have any `for-loops` in these functions. Include your code in the report.

     Important note: carefully read the provided code in `penalized_logistic_regression.R`. *You should understand the code and its structure instead of using it as a black box!*

3. (**2 pts**) Complete the missing parts in function `Penalized_Logistic_Reg` located at `penalized_logistic_regression.R`. This function should train the penalized logistic regression model using gradient descent on given training set. You may use the implemented functions from step 2. Include your code in the report.

   For parts 2 and 3, your completed `penalized_logistic_regression.R` *should NOT import other* `R` *packages.*

4. (**3 pts**) Complete the part (a) in `Q3_starter.R`.

   In this part, you need to fix your regularization parameter, `lbd = 0`, and to experiment with the hyperparameters for `stepsize` (the learning rate) and `max_iter` (the number of iterations).

   [Hints: (1) You only need to use the training data for this part. (2) A too small learning rate takes longer to converge. (3) A too large learning rate is also problematic.]

   - In the write-up, report and briefly explain which hyperparameter settings you found work the best.
   - For this choice of hyperparameters, generate and report a plot that shows how the training loss changes (iteration counter on x-axis and training loss on y-axis).
   - For this choice of hyperparameters, generate and report a plot for the training 0-1 error (iteration counter on x-axis and training error on y-axis).
   - Did the training 0-1 error have the same pattern as the training loss? Is your finding aligned with your expectation? State you reasoning.

5. (**4 pts**) Complete the part (b) in `Q3_starter.R`.

   Using the selected setting of hyperparameters (for learning rate and number of iteration) that you identified in step 4, fit the model by using $\lambda \in \{0, 0.01, 0.05, 0.1, 0.5, 1\}$.

   - (**1 pt**) Does your selected setting of hyperparameters guarantee convergence for all λ's? If not, re-identify hyperparameters for those λ's for which convergence is not guranteed. Report the hyperparameter setting(s) you used for each λ.
   - (**1 pt**) Generate and report one plot that shows how the training 0-1 error changes as you train with different values of λ.
   - (**1 pt**) Generate and report one plot that shows how the validation 0-1 error changes as you train with different values of λ.
   - (**1 pt**) Comment on the effects of λ based on these two plots. Which is the best value of λ based on your experiment?

6. (**2 pts**) Complete the part (c) in `Q3_starter.R`.

   Fit the model by using the best value of λ identified in step 5 and report its test 0-1 error. Compare your test error with the model fitted by using `glmnet` with the same λ.

7. (**2 pts**) Complete the part (d) in `Q3_starter.R`.

   Use your implementation of LDA and Naive Bayes in **Problem 2** to fit the training data, classify the test data and report the test 0-1 error.

8. (**2 pts**) Complete the part (e) in `Q3_starter.R`.

   Based on the test data, draw the ROC curves and compute the AUCs of your implemented penalized logistic regression, LDA and Naive Bayes. (You might find the package `roc` useful.)

SOLUTION:

1. We start by simplifying $\mathcal{J}(\boldsymbol{\beta}, \beta_0)$:

$$
\begin{aligned}
&\mathcal{J}(\boldsymbol{\beta}, \beta_0)\\
&= -\frac{1}{n}\sum_{i=1}^{n}\Big\{y_i \log\big[p(\boldsymbol{x}_i;\boldsymbol{\beta},\beta_0)\big] + (1-y_i)\log\big[1-p(\boldsymbol{x}_i;\boldsymbol{\beta},\beta_0)\big]\Big\} + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2\\
&= -\frac{1}{n}\sum_{i=1}^{n}\Big[y_i \log\Big(\frac{e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}}{1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}}\Big) + (1-y_i)\log\Big(\frac{1}{1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}}\Big)\Big] \;+\; \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2\\
&= -\frac{1}{n}\sum_{i=1}^{n}\Big[y_i\big[\log\big(e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}\big) - \log\big(1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}\big)\big] + (1-y_i)\big[-\log\big(1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}\big)\big)\big]\Big] + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2\\
&= -\frac{1}{n}\sum_{i=1}^{n}\Big[y_i \log\big(e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}\big) - \log\big(1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}\big)\big)\Big] + \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2\\
&= -\frac{1}{n}\sum_{i=1}^{n} y_i\big(\beta_0 + \boldsymbol{x}_i^\top\boldsymbol{\beta}\big) \;+\; \frac{1}{n}\sum_{i=1}^{n}\log\big(1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}\big) \;+\; \frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2
\end{aligned}
$$

Now we take a derivative with respect to $\boldsymbol{\beta}$:

$$
\frac{\partial \mathcal{J}(\boldsymbol{\beta},\beta_0)}{\partial \boldsymbol{\beta}} = -\frac{1}{n}\sum_{i=1}^{n} y_i\boldsymbol{x}_i + \frac{1}{n}\sum_{i=1}^{n}\frac{\boldsymbol{x}_i e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}}{1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}} + \lambda\boldsymbol{\beta}
$$

Plugging in specific $(\bar{\boldsymbol{\beta}}, \bar{\beta}_0)$,

$$
\frac{\partial \mathcal{J}(\boldsymbol{\beta},\beta_0)}{\partial \boldsymbol{\beta}}\Big|_{\bar{\boldsymbol{\beta}},\bar{\beta}_0} = \frac{1}{n}\sum_{i=1}^{n}\left[-y_i + \frac{e^{\bar{\beta}_0+\boldsymbol{x}_i^\top\bar{\boldsymbol{\beta}}}}{1+e^{\bar{\beta}_0+\boldsymbol{x}_i^\top\bar{\boldsymbol{\beta}}}}\right]\boldsymbol{x}_i + \lambda\bar{\boldsymbol{\beta}}
$$

For deriving $\frac{\partial \mathcal{J}(\boldsymbol{\beta},\beta_0)}{\partial \beta_0}\big|_{\bar{\boldsymbol{\beta}},\bar{\beta}_0}$, we can apply all of the simplification steps prior to taking the derivative that we just did to get the following:

$$
\begin{aligned}
\frac{\partial \mathcal{J}(\boldsymbol{\beta},\beta_0)}{\partial \beta_0} &= \frac{-\frac{1}{n}\sum_{i=1}^{n}\partial\Big[y_i\big(\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}\big)\Big]}{\partial\beta_0} + \frac{\frac{1}{n}\sum_{i=1}^{n}\partial\Big[\log\big(1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}\big)\Big]}{\partial\beta_0} + \frac{\partial\Big[\frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2\Big]}{\partial\beta_0}\\
&= -\frac{1}{n}\sum_{i=1}^{n} y_i + \frac{1}{n}\sum_{i=1}^{n}\frac{e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}}{1+e^{\beta_0+\boldsymbol{x}_i^\top\boldsymbol{\beta}}}
\end{aligned}
$$

Once again, we plug in specific specific $(\bar{\boldsymbol{\beta}}, \bar{\beta}_0)$:

$$
\frac{\partial \mathcal{J}(\boldsymbol{\beta},\beta_0)}{\partial \beta_0}\Big|_{\bar{\boldsymbol{\beta}},\bar{\beta}_0} = \frac{1}{n}\sum_{i=1}^{n}\left[-y_i + \frac{e^{\bar{\beta}_0+\boldsymbol{x}_i^\top\bar{\boldsymbol{\beta}}}}{1+e^{\bar{\beta}_0+\boldsymbol{x}_i^\top\bar{\boldsymbol{\beta}}}}\right]
$$

2. For the remaining parts of the problem, we have separate coding solutions