Problem Set III

1. **Bayesian Logistic Regression For Image Classification** **(**6501/6782−55%**,** 4501−80%**).** Given data (X, Y) with $X \in \mathbb{R}^d$ and $Y \in \{0, 1\}$, our goal is to train a classifier that will predict an unknown class label $\tilde{y}$ from a new data point $\tilde{x}$. Consider the following model:

$$Y \sim \text{Ber}\left(\frac{1}{1 + e^{-X^T\beta}}\right),$$
$$\beta \sim \text{N}(0, \sigma^2 I), \text{ where } I \text{ is an identity matrix.}$$

   This is a **Bayesian logistic regression** model. Your goal is to derive and implement a Bayesian inference on the coefficient β.

   (a) Write down the formula for the unormalized posterior of $\beta \mid Y$, i.e.,

$$p(\beta \mid y; x, \sigma) \propto \prod_{i=1}^{n} p(y_i \mid \beta; x_i) p(\beta; \sigma)$$

   (b) Show that this posterior is proportional to $\exp\left(-U(\beta)\right)$, where

$$U(\beta) = \sum_{i=1}^{n} (1 - y_i) x_i^T \beta + \log(1 + e^{-x_i^T \beta}) + \frac{1}{2\sigma^2} \|\beta\|^2.$$

   (c) Implement maximum a posterior (MAP) to infer β.

   (d) Use your code to analyze the minist dataset (provided in "mnist.mat"), looking only two digits 0 and 1. The 0/1 labels are your Y data, and the images are your X data. Also, add a constant term, i.e., a column of 1's to your X matrix. Make sure to train β from the provided training data, and then use the trained β to get a prediction, $\tilde{y}$, of the class labels for the test data.

   (e) Compare this to the true class labels, y, and see how well you did by estimating the average error rate, $\text{E}[|y - \tilde{y}|]$ (a.k.a. the zero-one loss). What values of σ and the MAP stepsize ϵ did you use?

   (f) Rerun your code to analyze the other two digits 6 and 8. Make sure to fine-tune the model parameters for best performance. Report your results of step (e).

2. (20%) Use the PCA algorithm (built-in Python function is allowed) to project the original images (only look at digits 0 and 1) to a low-dimensional space, and re-run the classification task above. Report your classification accuracy (and time consumption) with selected number of projected principal components ($PC = 10, 20, 30$).

3. (CS 6501 / ECE 6782 ONLY) **Image Augmentation** **(**25%**).** Implement image rotation (with angle $\theta \in [-15°, 15°]$) and translation (with randomly generated translation $T \in [-5, 5]$). Use these transformations to augment your limited training dataset (10% of the provided data for each class) and rerun your code on classifying digits 6 and 8. Report your classification accuracy with different number of augmented data (i.e., 3 times, 5 times of the training images per class) and discuss how does your augmentation affect the final results.

   *Note that Python built-in functions for image rotation and translation are allowed.