ISYE 6740 Homework 4

Total 100 points. 30 points Bonus questions.

1. **Basic optimization.** (50 points.)

   The background of logistic regression will be discussed in the next lecture. Here, we just focus on finding out the property of the optimization problem, related to training a logistic regression.

   Consider a simplified logistic regression problem. Given n training samples (x_i, y_i), $i = 1, \ldots, n$. The data $x_i \in \mathbb{R}$, and $y_i \in \{0, 1\}$. To train/fit a logistic regression model for classification, we solve the following optimization problem, where $\theta \in \mathbb{R}$ is a parameter we aim to find:

   $$\max_{\theta} \ell(\theta), \tag{1}$$

   where the log-likelihood function

   $$\ell(\theta) = \sum_{i=1}^{n} \left\{ -\log(1 + \exp\{-\theta x_i\}) + (y_i - 1)\theta x_i \right\}.$$

   (a) (15 points)

   Logistic regression model:

   $$p(y = 1|x, \theta) = \frac{1}{1 + \exp(-\theta^T x)}$$

   Note that:

   $$p(y = 0|x, \theta) = 1 - \frac{1}{1 + \exp(-\theta^T x)} = \frac{\exp(-\theta^T x)}{1 + \exp(-\theta^T x)}$$

   We are using maximum likelihood so plug in:

   $$\max_{\theta} \ell(\theta) := log \prod_{i=1}^{m} P(y^i|x^i, \theta)$$

   $$= \sum (y^i - 1)\theta^T x^i - log(1 + \exp(-\theta^T x^i))$$

   This is an unconstrined optimization problem, solvable using gradient descent.

   ```
   theta_t = inf
   theta_t_1 = rand
   error = .01
   while abs(theta_t - theta_t_1) > error:
           theta_t = theta_t_1
           theta_t_1 = get_negative_gradient(theta_t_1)
   ```

   (b) (15 points) **stochastic gradient descent** For huge datasets it may not be feasible to use gradient descent. Batch gradient descent can be used to overcome this obstacle by randomly selecting a subset of data and applying gradient descent and eventually looping through all of the data. The gradient is unbiased so the expectation equals the true gradient.

$$\nabla\hat{\ell}(\theta) = \sum_{y^i \epsilon S_k} (y^i - 1)x^i + \frac{\exp(-\theta^T x^i)x^i}{1 + \exp(-\theta^T x^i)}$$

(c) (20 points) If a function f is differentiable it can be chatacterized as concave if the negative of the function (that is if f is concave $-f$ is convex) follows the first and second order conditions resulting in a positive semi-definite Hessian.
First order condition:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x)$$

for all $x, y \epsilon \text{dom} f$
Second Order condition: f is convex iff $\text{dom} f$ is convex and for all $x \epsilon \text{dom} f$

$$\nabla^2 f(x) \geq 0$$

Solution:

$$l(\theta) = \sum_{i=1}^{n} \{-log\left(1 + e^{-\theta x_i}\right) + (y_i - 1)\,\theta x_i\}$$

$$\begin{bmatrix} \frac{\partial^2 l}{\partial \theta_1^2} & \cdots & \frac{\partial^2 l}{\partial \theta_1 \partial \theta_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 l}{\partial \theta_n \partial \theta_1} & \cdots & \frac{\partial^2 l}{\partial \theta_n^2} \end{bmatrix}$$

$$\left[\frac{\partial^2 l}{\partial \theta^2}\right] = -\frac{x^2 \exp(x\theta)}{(\exp(x\theta) + 1)^2}$$

Since we see that the above is indeed a concave function we know that there is a single global maximum and is a great candidate for gradient descent. Gradient descent travels along the function's gradient in pre defined learning steps until it reaches the maximum. Since we know that there are not any local maxima once the maximum is found training is complete.

2. **Comparing Bayes, logistic, and KNN classifiers.** (50 points)

In lectures we learn three different classifiers. This question is to implement and compare them. We are suggest use Scikit-learn, which is a commonly-used and powerful Python library with various machine learning tools. But you can also use other similar library in other languages of your choice to perform the tasks.

Part One (Divorce classification/prediction). (30 points)

This dataset is about participants who completed the personal information form and a divorce predictors scale.

The data is a modified version of the publicly available at `https://archive.ics.uci.edu/ml/datasets/Divorce+Predictors+data+set` (by injecting noise so you will not replicate the results on uci website). There are 170 participants and 54 attributes (or predictor variables) that are all real-valued. The dataset **q3.csv**. The last column of the CSV file is label y (1 means "divorce", 0 means "no divorce"). Each column is for one feature (predictor variable), and each row is a sample (participant). A detailed explanation for each feature (predictor variable) can be found at the website link above. Our goal is to build a classifier using training data, such that given a test sample, we can classify (or essentially predict) whether its label is 0 ("no divorce") or 1 ("divorce").

Build three classifiers using (Naive Bayes, Logistic Regression, KNN). Use the first 80% data for training and the remaining 20% for testing. If you use scikit-learn you can use train_test_split to split the dataset.

(a) Report testing accuracy for each of the three classifiers. Comment on their performance: which performs the best and make a guess why they perform the best in this setting.

(b) Use the first two features to train three new classifiers. Plot the data points and decision boundary of each classifier. Comment on the difference between the decision boundary for the three classifiers. Please clearly represent the data points with different labels using different colors.

Part Two (Handwritten digits classification). (20 points) Repeat the above using the **MNIST Data** in our **Homework 3**. Here, give "digit" 6 label $y = 1$, and give "digit" 2 label $y = 0$. All the pixels in each image will be the feature (predictor variables) for that sample (i.e., image). Our goal is to build classifier to such that given a new test sample, we can tell is it a 2 or a 6. Using the first 80% of the samples for training and remaining 20% for testing. Report the classification accuracy on testing data, for each of the three classifiers. Comment on their performance: which performs the best and make a guess why they perform the best in this setting.

3. **Deriving M-step in EM for GMM.** (Bonus question: 10 points)

Consider the Q function in EM for GMM:

$$Q(\theta|\theta_k) = \sum_{i=1}^{n}\sum_{c=1}^{C} p_{i,c} \log \pi_c + \sum_{i=1}^{n}\sum_{c=1}^{C} p_{i,c} \log \phi(x_i|\mu_c, \Sigma_c)$$

where $\theta = \{\pi_c, \mu_c, \Sigma_c\}_{c=1,\ldots,C}$, and $p_{i,c} \propto \pi_c^{(k)} \phi(x_i|\mu_c^{(k)}, \Sigma_c^{(k)})$, $i = 1, \ldots, n$, $c = 1, \ldots, C$ depend on the parameters from the previous iteration. Here $\phi(x|\mu, \Sigma)$ denotes the pdf of a multi-variate Gaussian with mean vector μ and covariance matrix Σ.

Solve π_c, μ_c and Σ_c that maximize the $Q(\theta|\theta_k)$ function. In other words, we want to find

$$\begin{aligned} \theta_{k+1} := \arg\max_{\theta} \quad & Q(\theta|\theta_k) \\ \text{subject to} \quad & \sum_{c=1}^{C} \pi_c = 1. \end{aligned}$$

Show that the solution θ_{k+1} is given by the following expression

$$\pi_c^{(k+1)} = \frac{\sum_{i=1}^{n} p_{i,c}}{n}$$

$$\mu_c^{(k+1)} = \frac{\sum_{i=1}^{n} p_{i,c} x_i}{\sum_{i=1}^{c} p_{i,c}}$$

$$\Sigma_c^{(k+1)} = \frac{\sum_{i=1}^{n} p_{i,c} (x_i - \mu_c^{(k)})(x_i - \mu_c^{(k)})^T}{\sum_{i=1}^{n} p_{i,c}}$$

(Hint: Consider the Lagrangian function for solving this constrained optimization problem. You only need to introduce one Lagrangian multiplier because you only have one constraint. Then solve it from there.)

4. **Naive Bayes for spam filtering**. (Bonus question: 20 points)

In this problem we will use the Naive Bayes algorithm to fit a spam filter by hand. This will enhance your understanding to Bayes classifier and build intuition.

Spam filters are used in all email services to classify received emails as "Spam" or "Not Spam". A simple approach involves maintaining a vocabulary of words that commonly occur in "Spam" emails

and classifying an email as "Spam" if the number of words from the dictionary that are present in the email is over a certain threshold. We are given the vocabulary consists of 15 words

$V = \{\text{secret, offer, low, price, valued, customer, today, dollar, million, sports, is, for, play, healthy, pizza}\}$.

We will use V_i to represent the ith word in V. As our training dataset, we are also given 3 example spam messages,

- million dollar offer
- secret offer today
- secret is secret

and 4 example non-spam messages

- low price for valued customer
- play secret sports today
- sports is healthy
- low price pizza

Recall that the Naive Bayes classifier assumes the probability of an input $x = [x_1, x_2, \ldots, x_n]^T$ depends on its class y. In our case the input vector x corresponding to each message has length $n = 15$ equal to the number of words in the vocabulary V, where each entry x_i is equal to the number of times word V_i occurs in x.

(a) Calculate $\mathbb{P}(y = 0)$ and $\mathbb{P}(y = 1)$ from the training data, where $y = 0$ corresponds to spam messages, and $y = 1$ corresponds to non-spam messages.

(b) List the feature vector x for each spam and non-spam message.

(c) In the Naive Bayes model, the likelihood of a sentence with feature vector x given a class c is

$$\mathbb{P}(x|y = c) = \prod_{k=1}^{n} \theta_{c,k}^{x_k}$$

where $\theta_{c,k} \in (0, 1)$ is the weight of word k in class c, which satisfies $\sum_{k=1}^{n} \theta_{c,k} = 1$, $\forall c$. Calculate the maximum likelihood estimates of $\theta_{0,1}$, $\theta_{0,7}$, $\theta_{1,1}$, $\theta_{1,15}$ by maximizing $\mathbb{P}(x|y = c)$ with respect to $\theta_{c,k}$ and given data. (Hint: Consider the Lagrangian function for solving this constrained optimization problem. You only need to introduce one Lagrangian multiplier because you only have one constraint. Consider log-likelihood (i.e., taking log of the cost function). Then solve it from there.)

(d) Given a new message "today is secret", decide whether it is spam or not spam, based on the Naive Bayes classifier, learned from the above data.