EECS545 Machine Learning (WN 2021)
Practice Midterm Exam Solution

The midterm is worth 30% of your final grade in this course. You may use your book and notes to complete the exam. However, the use of any electronic devices, including phones and laptops are strictly prohibited. You are also not allowed to discuss any portion of the exam with any other student while the exam is being proctored. Your objective is to receive as many points out of 100 as possible.

Name: _____

Uniqname: _____

I attest that I have not given or received aid in this examination, and that I have done my share and taken an active part in seeing to it that others as well as myself uphold the spirit and letter of the University of Michigan College of Engineering Honor Code.

Specifically for this examination, I attest that I have not used a laptop, a smartphone, any electronic device, or any online materials.

Signed: _____

Question	Points
1. Robust Linear Regression	/12
2. Cost-sensitive SVM	/12
3. Logistic Regression with Noisy Labels	/18
4. Kernelizing Ridge Regression	/12
5. Short Answers	/46
Total	/100

1 [12 points] Robust linear regression

Consider the training data $\{(\mathbf{x}^{(1)}, t^{(1)}), \cdots, (\mathbf{x}^{(N)}, t^{(N)})\}$ where $\mathbf{x} \in \mathbb{R}^D$ and $t \in \mathbb{R}$. Assume that the output t is generated from input $\mathbf{x}$ as follows:

$$\begin{aligned} t &= \mathbf{w}^T \phi(\mathbf{x}) + \epsilon \\ \epsilon &\sim \text{Laplace}(\epsilon; 0, 1) \end{aligned}$$

where the probability density function of Laplace distribution is given as

$$\text{Laplace}(x; \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x - \mu|}{b}\right)$$

(a) [4 points] Show that the Maximum Likelihood Estimation (MLE) of $\mathbf{w}$ for the data (i.e., maximizing the log-likelihood of t conditioned on $\mathbf{x}$ over the training data) is equivalent to the "robust linear regression" problem, which is written as

$$\min_{\mathbf{w}} \sum_{i=1}^{N} |t^{(i)} - \mathbf{w}^T \phi(\mathbf{x}^{(i)})|$$

ANSWER:

The log-likelihood is written as follows:

$$\begin{aligned} \frac{1}{N} \sum_{i=1}^{N} \log P\left(t^{(i)} | \mathbf{x}^{(i)}\right) &= \frac{1}{N} \sum_{i=1}^{N} \log P\left(t^{(i)} | \mathbf{x}^{(i)}\right) \ldots \text{[1 point]} \\ &= \frac{1}{N} \sum_{i=1}^{N} \log \frac{1}{2} \exp\left(-\left|t^{(i)} - \mathbf{w}^T \phi\left(\mathbf{x}^{(i)}\right)\right|\right) \ldots \text{[1 point]} \end{aligned}$$

Therefore, the MLE of t conditioned on $\mathbf{x}$ can be written as

$$\begin{aligned} \hat{\mathbf{w}} &= \arg\max_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} \log P\left(t^{(i)} | \mathbf{x}^{(i)}\right) \ldots \text{[1 point]} \\ &= \arg\max_{\mathbf{w}} \sum_{i=1}^{N} -\left|t^{(i)} - \mathbf{w}^T \phi\left(\mathbf{x}^{(i)}\right)\right| \ldots \text{[1 point]} \\ &= \arg\min_{\mathbf{w}} \sum_{i=1}^{N} \left|t^{(i)} - \mathbf{w}^T \phi\left(\mathbf{x}^{(i)}\right)\right| \end{aligned}$$

(b) [4 points] Derive the gradient of the negative log-likelihood with respect to $\mathbf{w}$. For sake of simplicity, you may use a specific form of subgradient:[1] $\frac{\partial |s|}{\partial s} = sign(s) = \begin{cases} 1 & \text{if } s > 0 \\ 0 & \text{if } s = 0 \\ -1 & \text{if } s < 0 \end{cases}$.

ANSWER: The gradient can be computed as

$$\begin{aligned} \frac{\partial}{\partial \mathbf{w}} - \frac{1}{N}\sum_{i=1}^{N} \log P\left(t^{(i)}|\mathbf{x}^{(i)}\right) &= \frac{1}{N}\sum_{i=1}^{N} \frac{\partial \left|t^{(i)} - \mathbf{w}^T\phi\left(\mathbf{x}^{(i)}\right)\right|}{\partial \mathbf{w}} \ldots \text{[2 point]} \\ &= \frac{1}{N}\sum_{i=1}^{N} sign\left(t^{(i)} - \mathbf{w}^T\phi\left(\mathbf{x}^{(i)}\right)\right) \frac{\partial \left(t^{(i)} - \mathbf{w}^T\phi\left(\mathbf{x}^{(i)}\right)\right)}{\partial \mathbf{w}} \ldots \text{[2 point]} \\ &= -\frac{1}{N}\sum_{i=1}^{N} sign\left(t^{(i)} - \mathbf{w}^T\phi\left(\mathbf{x}^{(i)}\right)\right) \phi\left(\mathbf{x}^{(i)}\right) \end{aligned}$$

(c) [4 points] What would be the pros and cons of this version of linear regression if we compare with the least squares linear regression (i.e., with L_2 penalty) that is discussed in the class?

ANSWER:

[2 points] Pros: The robust linear regression is more robust to the outliers (examples whose target value largely deviate from the predictions) than least square linear regression.. . . [2 point]

Error cases:

- robust to overfitting.
- better fit when the noise is distributed with Laplcian distribution (vague; some deduction).

[2 points] Cons: The robust linear regression is more difficult to optimize. (Due to the non-differentiable objective function, simply using gradient descent will make the solution "oscillate" and converge very slowly.). . . [2 point]

[1] Intuitively speaking, subgradient is often used when the objective function is convex, but not differentiable. A rigorous treatment of subgradient is beyond the scope of this course.

2 [12 points] Implementing Cost-sensitive Soft Margin SVM by Optimizing Primal Objective

In the lecture, we introduced SVM with the following constrained objective function:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{N}\xi^{(i)} \\ \text{subject to} \quad & t^{(i)}(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi^{(i)},\ i = 1,\ldots,N \\ & \xi^{(i)} \geq 0,\ i = 1,\ldots,N \end{aligned}$$

Now we will modify this model so that it values the misclassified example from the two classes differently. Instead, we have the objective function as:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 + C_+\sum_{i:t^{(i)}=1}\xi^{(i)} + C_-\sum_{i:t^{(i)}=-1}\xi^{(i)} \\ \text{subject to} \quad & t^{(i)}(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi^{(i)},\ i = 1,\ldots,N \\ & \xi^{(i)} \geq 0,\ i = 1,\ldots,N \end{aligned}$$

(a) **[6 points]** Find the derivatives of the loss function with respect to our parameters. Show that:

$$\nabla_{\mathbf{w}}E(\mathbf{w},b) = \mathbf{w} - C_+\sum_{i:t^{(i)}=1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) < 1\right]\phi(\mathbf{x}^{(i)}) + C_-\sum_{i:t^{(i)}=-1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) > -1\right]\phi(\mathbf{x}^{(i)})$$

$$\frac{\partial}{\partial b}E(\mathbf{w},b) = -C_+\sum_{i:t^{(i)}=1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) < 1\right] + C_-\sum_{i:t^{(i)}=-1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) > -1\right]$$

ANSWER:

The objective function can be combined with the constraints and becomes:

$$J(\mathbf{w},b) = \frac{1}{2}\|\mathbf{w}\|^2 + C_+\sum_{i:t^{(i)}=1}\max\left\{0, 1 - \left(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b\right)\right\} + C_-\sum_{i:t^{(i)}=-1}\max\left\{0, 1 + \left(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b\right)\right\} \ldots \text{[2 point]}$$

With that, the derivatives w.r.t. each of the parameters are:

$$\begin{aligned} \nabla_{\mathbf{w}}E(\mathbf{w},b) &= \nabla_{\mathbf{w}}J(\mathbf{w},b) \\ &= \mathbf{w} - C_+\sum_{i:t^{(i)}=1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) \geq 1\right]\mathbf{0} - C_+\sum_{i:t^{(i)}=1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) < 1\right]\phi(\mathbf{x}^{(i)}) \\ &\quad + C_-\sum_{i:t^{(i)}=-1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) \leq -1\right]\mathbf{0} + C_-\sum_{i:t^{(i)}=-1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) > -1\right]\phi(\mathbf{x}^{(i)}) \\ &= \mathbf{w} - C_+\sum_{i:t^{(i)}=1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) < 1\right]\phi(\mathbf{x}^{(i)}) + C_-\sum_{i:t^{(i)}=-1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) > -1\right]\phi(\mathbf{x}^{(i)}) \ldots \text{[2 point]} \end{aligned}$$

Similarily,

$$\frac{\partial}{\partial b}E(\mathbf{w},b) = -C_+\sum_{i:t^{(i)}=1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) < 1\right] + C_-\sum_{i:t^{(i)}=-1}\mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) > -1\right] \ldots \text{[2 point]}$$

(b) **[6 points]** Suppose that a group of students in EECS 545 are working on a project which tries to solve the fraud detection problem. They are given a set of a particular user's legitimate credit card transactions, each has the same set of features including location, amount, date, etc. of the transaction, as the training data, and some pending transactions with the same set of features as the test data. In testing phase, they need to classify each pending transaction into two classes, legitimate transaction or fraudulent transaction. The problem is that there is no training example from the fraudulent transaction class. Please propose a modification of the model from part (a) to accommodate this situation and show the training objective function, the update rule for stochastic gradient training, and the classification function (i.e., how you would detect suspicious transactions in testing time).

ANSWER:

This problem is called novelty detection problem. It is a weakly supervised problem because in this problem all training data are implicitly labeled as from the same class, in our case, legitimate transaction. Consequently, there is no training data from the other class which in our case is suspicious transaction. To solve this problem, we will take advantage of the model we just developed. With some thoughts it is not hard to find out that having no training data from one class which we can safely assume to be the class with label of -1, is equivalent to setting the penalizing parameter of that class, C_- to be 0 which will result in an adjusted model with the objective function of:

$$J(\mathbf{w}, b) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i}^{N} \max\left\{0, 1 - \left(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b\right)\right\} \ldots \text{[2 point]}$$

The derivative of the parameters will be:

$$\nabla_{\mathbf{w}} E(\mathbf{w}, b) = \mathbf{w} - C\sum_{i=1}^{N} \mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) < 1\right]\phi(\mathbf{x}^{(i)}) \ldots \text{[1 point]}$$

$$\frac{\partial}{\partial b} E(\mathbf{w}, b) = -C\sum_{i=1}^{N} \mathbf{I}\left[(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) + b) < 1\right] \ldots \text{[1 point]}$$

The classification function will remain the same:

$$t^{test} = sign\left(\mathbf{w}^T\phi(\mathbf{x}^{test}) + b\right) \ldots \text{[2 point]}$$

where

$$sign(x) = \begin{cases} 1 & \text{, if } x \geq 0 \\ -1 & \text{, if } x < 0 \end{cases}$$

3 [18 points] Logistic Regression with Noisy Labels

So far in our classification experiments we have always assumed that the training data are correctly labeled. However sometimes this is not the case. Now assume that because your GSI has a bad eyesight and that he sometimes misread the class label. Unfortunately he has no clue whatsoever how often does he misread a class label when constructing the dataset. Therefore in the training data, we have $\{(\mathbf{x}^{(1)}, t^{(1)}), (\mathbf{x}^{(2)}, t^{(2)}), \ldots, (\mathbf{x}^{(N)}, t^{(N)})\}$, where $t \in \{0, 1\}$, the labels are not that reliable (fortunately all the $\mathbf{x}$ are correctly collected) and the probability of any of the data being mislabeled is ϵ which is known. (In general, this ϵ could be an unknown parameter to estimate via EM, but we assume that it's given for the sake of simplicity.) To address this problem, we are going to use the Expectation Maximization method, where we treat true label $z^{(i)}$ as a latent variable corresponding to each noisy label $t^{(i)}$. Specifically,

$$P(t^{(i)}|z^{(i)}) = \begin{cases} 1-\epsilon & \text{, if } t^{(i)} = z^{(i)} \\ \epsilon & \text{, if } t^{(i)} \neq z^{(i)} \end{cases}$$

In other words, we assume that there is a small probability ϵ that true label $z^{(i)}$ and noisy label $y^{(i)}$ do not agree. We use the same logistic regression formulation as usual, i.e.,

$$P(z^{(i)} = 1|\mathbf{x}^{(i)}) = \sigma\left(\mathbf{w}^T\phi(\mathbf{x}^{(i)})\right)$$
$$P(z^{(i)} = 0|\mathbf{x}^{(i)}) = 1 - \sigma\left(\mathbf{w}^T\phi(\mathbf{x}^{(i)})\right)$$

where $\sigma(s) = \frac{1}{1+\exp(-s)}$ is a logistic sigmoid function.

(a) [1 point] Please explain, in 1-2 sentences, why the EM algorithm is an appropriate choice to solve this problem.

ANSWER: The problem has the ground truth labels of the training data remain unknown. However for each training point, the expectation of the ground truth can be inferred using the given information. With the expected ground truth value we can train a classification model by some optimization algorithm. This pipeline fits well into the scheme of EM algorithm. ... [1 point]
Note: any reasonable explanation can get full credit.

(b) [4 points] To apply the EM algorithm, first write out the log-likelihood function, and show that it can be lower bounded as follows. (Hint: note that $P(t^{(i)}|z^{(i)}) = P(t^{(i)}|z^{(i)}, \mathbf{x}^{(i)})$ because once $z^{(i)}$ is given, $t^{(i)}$ is simply determined by "coin flip" with probability ϵ, which is not affected by $\mathbf{x}^{(i)}$.) Below, note that the inequality holds true for any valid Bernoulli distribution $Q(z^{(i)})$, but it is tight when $Q(z^{(i)}) = P(z^{(i)}|\mathbf{x}^{(i)}, t^{(i)})$, which will be calculated in E-step derivation.

$$\sum_{i=1}^{N} \log P(t^{(i)}|\mathbf{x}^{(i)}) \geq \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \log \frac{P(t^{(i)}|z^{(i)})P(z^{(i)}|\mathbf{x}^{(i)})}{Q(z^{(i)})}$$

ANSWER:

$$\begin{aligned}
\sum_{i=1}^{N} \log P(t^{(i)}|x^{(i)}) &= \sum_{i=1}^{N} \log \sum_{z^{(i)}\in\{0,1\}} P(t^{(i)}|x^{(i)}, z^{(i)}) P(z^{(i)}|x^{(i)}) \ldots \text{[1 point]} \\
&= \sum_{i=1}^{N} \log \sum_{z^{(i)}\in\{0,1\}} Q(z^{(i)}) \frac{P(t^{(i)}|x^{(i)}, z^{(i)}) P(z^{(i)}|x^{(i)})}{Q(z^{(i)})} \ldots \text{[1 point]} \\
&\geq \sum_{i=1}^{N} \sum_{z^{(i)}\in\{0,1\}} Q(z^{(i)}) \log \frac{P(t^{(i)}|x^{(i)}, z^{(i)}) P(z^{(i)}|x^{(i)})}{Q(z^{(i)})} \ldots \text{[2 point]}
\end{aligned}$$

(c) [3 points] Derive E-step for the EM algorithm. First, show that:

$$\begin{aligned}
P(t=1|\mathbf{x}) &= (1-\epsilon)\sigma(\mathbf{w}^T\phi(\mathbf{x})) + \epsilon\left(1-\sigma(\mathbf{w}^T\phi(\mathbf{x}))\right) \\
P(t=0|\mathbf{x}) &= \epsilon\sigma(\mathbf{w}^T\phi(\mathbf{x})) + (1-\epsilon)\left(1-\sigma(\mathbf{w}^T\phi(\mathbf{x}))\right)
\end{aligned}$$

therefore, the posterior of the true label given the input and its noisy label $(x^{(i)}, t^{(i)})$ can be computed as follows:

$$\begin{aligned}
P(z|x, t=1) &= \frac{\left[(1-\epsilon)\sigma(\mathbf{w}^T\phi(\mathbf{x}))\right]^{I(z=1)}\left[\epsilon\left(1-\sigma(\mathbf{w}^T\phi(\mathbf{x}))\right)\right]^{I(z=0)}}{(1-\epsilon)\sigma(\mathbf{w}^T\phi(\mathbf{x})) + \epsilon\left(1-\sigma(\mathbf{w}^T\phi(\mathbf{x}))\right)} \\
P(z|x, t=0) &= \frac{\left[\epsilon\sigma(\mathbf{w}^T\phi(\mathbf{x}))\right]^{I(z=1)}\left[(1-\epsilon)\left(1-\sigma(\mathbf{w}^T\phi(\mathbf{x}))\right)\right]^{I(z=0)}}{\epsilon\sigma(\mathbf{w}^T\phi(\mathbf{x})) + (1-\epsilon)\left(1-\sigma(\mathbf{w}^T\phi(\mathbf{x}))\right)}
\end{aligned}$$

ANSWER: *Note: you can get full credit if you only derive $P(z|x,t)$ and get it correctly.*
The key idea is to note that $P(t|x) = \sum_z P(t,z|x) = \sum_z P(t|z,x)P(z|x) = \sum_z P(t|z)P(z|x)$

$$\begin{aligned}
P(t=1|\mathbf{x}) &= \sum_z P(z|\mathbf{x})P(t=1|\mathbf{x},z) \\
&= P(z=1|\mathbf{x})P(y=1|\mathbf{x},z=1) + P(z=0|\mathbf{x})P(t=1|\mathbf{x},z=0) \\
&= P(z=1|\mathbf{x})P(t=1|z=1) + P(z=0|\mathbf{x})P(t=1|z=0) \\
&= (1-\epsilon)P(z=1|\mathbf{x}) + \epsilon P(z=0|\mathbf{x}) \\
&= (1-\epsilon)\sigma(w^T\phi(\mathbf{x})) + \epsilon\left(1-\sigma(w^T\phi(\mathbf{x}))\right) \ldots \text{[0.5 point]}
\end{aligned}$$

Similarly,

$$P(t=0|\mathbf{x}) = \epsilon\sigma(w^T\phi(\mathbf{x})) + (1-\epsilon)\left(1-\sigma(w^T\phi(\mathbf{x}))\right) \ldots \text{[0.5 point]}$$

E-step:

$$\begin{aligned}
P(z|\mathbf{x}, t=1) &= \frac{P(t=1,z|\mathbf{x})}{P(t=1|\mathbf{x})} \\
&= \frac{\left[(1-\epsilon)\sigma(w^T\phi(\mathbf{x}))\right]^{I(z=1)}\left[\epsilon\left(1-\sigma(w^T\phi(\mathbf{x}))\right)\right]^{I(z=0)}}{(1-\epsilon)\sigma(w^T\phi(\mathbf{x})) + \epsilon\left(1-\sigma(w^T\phi(\mathbf{x}))\right)} \ldots \text{[1 point]} \\
P(z|\mathbf{x}, t=0) &= \frac{P(t=0,z|\mathbf{x})}{P(t=0|\mathbf{x})} \\
&= \frac{\left[\epsilon\sigma(w^T\phi(\mathbf{x}))\right]^{I(z=1)}\left[(1-\epsilon)\left(1-\sigma(w^T\phi(\mathbf{x}))\right)\right]^{I(z=0)}}{\epsilon\sigma(w^T\phi(\mathbf{x})) + (1-\epsilon)\left(1-\sigma(w^T\phi(\mathbf{x}))\right)} \ldots \text{[1 point]}
\end{aligned}$$

(d) [4 points] Here, we will use $Q(z^{(i)}) = P(z^{(i)}|\mathbf{x}^{(i)}, t^{(i)})$ for the M-step. Derive M-step for the EM algorithm. Specifically, the M-step maximizes the lower bound with respect to model parameter θ (in this problem, we only consider the logistic regression parameter $\mathbf{w}$, so θ is $\mathbf{w}$). In other words, show that the new parameter update is given as follows:

$$\theta^{new} := \arg\max_{\theta} \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \log P_\theta(z^{(i)}|\mathbf{x}^{(i)})$$

ANSWER:

$$\begin{aligned}
\theta^{new}: &= \arg\max_{\theta} \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \log \frac{P(t^{(i)}|\mathbf{x}^{(i)}, z^{(i)}) P_\theta(z^{(i)}|\mathbf{x}^{(i)})}{Q(z^{(i)})} \ldots \text{[1 point]} \\
&= \arg\max_{\theta} \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \log P(t^{(i)}|\mathbf{x}^{(i)}, z^{(i)}) P_\theta(z^{(i)}|\mathbf{x}^{(i)}) \ldots \text{[1 point]} \\
&= \arg\max_{\theta} \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \log P_\theta(z^{(i)}|\mathbf{x}^{(i)}) \ldots \text{[2 point]}
\end{aligned}$$

For the last step, we used the fact that $P(t^{(i)}|\mathbf{x}^{(i)}, z^{(i)}) = P(t^{(i)}|z^{(i)}$ does not depend on $\mathbf{w}$.

(e) [4 points] To solve the M-step above, assume that we use gradient descent. Please compute the gradient of the lower-bound above.

ANSWER:

$$\begin{aligned}
\Delta\theta &= \frac{\partial}{\partial\theta} \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \log P_\theta(z^{(i)}|x^{(i)}) \\
&= \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \frac{\partial}{\partial\theta} \log P_\theta(z^{(i)}|x^{(i)}) \ldots \text{[2 point]}
\end{aligned}$$

This is basically weighted gradient of the negative log-likelihood. Specifically, we can reuse the basic result from the logistic regression (homework #1) for gradient calculation.

$$\begin{aligned}
& \frac{\partial}{\partial w} \log P_\theta(z^{(i)}|x^{(i)}) \\
= \; & \frac{\partial}{\partial w} \log \sigma^{z^{(i)}} (1-\sigma)^{(1-z^{(i)})} \\
= \; & \frac{\partial}{\partial w} \left[z^{(i)} \log\sigma + (1-z^{(i)}) \log(1-\sigma) \right] \\
= \; & z^{(i)}(1-\sigma)\phi(x^{(i)}) + (1-z^{(i)})(-\sigma)\phi(x^{(i)}) \\
= \; & (z^{(i)} - \sigma)\phi(x^{(i)}) \ldots \text{[1 point]}
\end{aligned}$$

Here, we used some shorthand notation, such as $\sigma = P_\theta(z^{(i)}|\mathbf{x}^{(i)}) = \sigma(\mathbf{w}^T\phi(\mathbf{x}^{(i)}))$. Putting things

together, we have:

$$\begin{aligned}
\Delta \mathbf{w} &= \frac{\partial}{\partial \mathbf{w}} \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \log P_\theta(z^{(i)} | \mathbf{x}^{(i)}) \\
&= \sum_{i=1}^{N} \sum_{z^{(i)} \in \{0,1\}} Q(z^{(i)}) \left(z^{(i)} - \sigma(\mathbf{w}^T \phi(\mathbf{x}^{(i)})) \right) \phi(\mathbf{x}^{(i)}) \\
&= \sum_{i=1}^{N} \left(Q(z^{(i)}) - \sigma(w^T \phi(\mathbf{x}^{(i)})) \right) \phi(\mathbf{x}^{(i)}) \ldots \text{[1 point]}
\end{aligned}$$

(f) [2 points] As another way of solving the M-step, suppose that you have a black-box logistic regression solver that can get example-wise weight as input. Describe how you would use this black-box logistic regression solver to solve the M-step.

ANSWER:

If you have a black-box logistic regression solver that can get example-wise weight as input, then you can just provide weights for the two possible true labels. I.e., use weight $Q(z^{(i)} = 1)$ for $(\mathbf{x}^{(i)}, z^{(i)} = 1)$, and $Q(z^{(i)} = 0)$ for $(\mathbf{x}^{(i)}, z^{(i)} = 0)$. .

4 [10 points] Kernelizing Ridge Regression

The objective function of Ridge Regression with L_2 regularizer is:

$$\frac{1}{2}\sum_{i=1}^{N}\left(t^{(i)} - \mathbf{w}^T\phi(\mathbf{x}^{(i)})\right)^2 + \frac{1}{2}\lambda\|\mathbf{w}\|^2$$

For simplicity in this problem we assume that both x and t are centered with 0 mean. With this assumption, the optimal parameters are:

$$w^* = (\Phi^T\Phi + \lambda\mathbf{I})^{-1}\Phi^T T$$

where

$$\Phi = \begin{bmatrix} \phi\left(\mathbf{x}^{(1)}\right)^T \\ \phi\left(\mathbf{x}^{(2)}\right)^T \\ \vdots \\ \phi\left(\mathbf{x}^{(N)}\right)^T \end{bmatrix}$$

$$T = \begin{bmatrix} t^{(1)} \\ t^{(2)} \\ \vdots \\ t^{(N)} \end{bmatrix}$$

(a) [4 points] Given the optimal parameter $\mathbf{w}^*$ above and a test data $\mathbf{x}^{test}$, what is the predicted t^{test}? Is this prediction function kernelized as it is? Why or why not?

ANSWER:

The classification function is:

$$y^{test} = T^T\Phi(\Phi^T\Phi + \lambda\mathbf{I})^{-1}\phi(\mathbf{x}^{test})$$

With this current form, it cannot be kernelized with simple substitutions since $\Phi^T\Phi$ is not a kernel (note that the Gram matrix is $\Phi\Phi^T$).
Note: you can get full credit if you find some other way to correctly kernelize the function. You can get partial credit if you only state that the function is not kernelized because there are Φ and $\phi(\mathbf{x})$ in it.

(b) [6 points] Given the matrix inverse lemma

$$(QRS + P)^{-1} = P^{-1} - P^{-1}Q(R^{-1} + SP^{-1}Q)^{-1}SP^{-1}$$

kernelize the Ridge Regression. Show the kernelized equation for t^{test} (where only kernel or gram matrix K shall appear but no Φ or $\phi(\mathbf{x}^{test})$ is used explicitly.)

Note: in the exam there was a typo in the lemma which is fixed here. There should be a minus sign instead of plus after the P^{-1} term on the right hand side. No penalty will be applied for any error as a consequence of this typo in the exam.

ANSWER:

Let $Q = \Phi^T,\ R = \mathbf{I},\ S = \Phi,\ P = \lambda\mathbf{I}\ldots$ [1 point], using the matrix inverse lemma,

$$\begin{aligned}
(\Phi^T\Phi + \lambda\mathbf{I})^{-1} &= (\lambda\mathbf{I})^{-1} - (\lambda\mathbf{I})^{-1}\Phi^T(\mathbf{I} + \Phi(\lambda\mathbf{I})^{-1}\Phi^T)^{-1}\Phi(\lambda\mathbf{I})^{-1} \\
&= \frac{1}{\lambda}\mathbf{I} - \frac{1}{\lambda}\Phi^T(\lambda\mathbf{I} + \Phi\Phi^T)^{-1}\Phi \\
w^* &= (\Phi^T\Phi + \lambda\mathbf{I})^{-1}\Phi^T T \\
&= \left(\frac{1}{\lambda}\mathbf{I} - \frac{1}{\lambda}\Phi^T(\lambda\mathbf{I} + \Phi\Phi^T)^{-1}\Phi\right)\Phi^T T \\
&= \frac{1}{\lambda}\left(\Phi^T - \Phi^T(\lambda\mathbf{I} + K)^{-1}K\right)T \ldots \text{[2 point]} \\
y^{test} &= w^{*T}\phi(\mathbf{x}^{test}) \\
&= \left(\frac{1}{\lambda}\Phi^T\left(\mathbf{I} - (\lambda\mathbf{I} + K)^{-1}K\right)T\right)^T\phi(\mathbf{x}^{test}) \\
&= \frac{1}{\lambda}T^T\left(\mathbf{I} - (\lambda\mathbf{I} + K)^{-1}K\right)^T\Phi\phi(\mathbf{x}^{test}) \\
&= \frac{1}{\lambda}T^T\left(\mathbf{I} - K\left(\lambda\mathbf{I} + \mathbf{K}\right)^{-1}\right)k(\mathbf{x}^{test}) \ldots \text{[2 point]}
\end{aligned}$$

where

$$\begin{aligned}
K &= \left[\phi\left(\mathbf{x}^{(i)}\right)^T\phi\left(\mathbf{x}^{(j)}\right)\right]_{i,j=1}^{N} \ldots \text{[0.5 point]} \\
k(\mathbf{x}) &= \left[\phi(\mathbf{x})^T\phi(\mathbf{x}^{(i)})\right]_{i=1}^{N} \ldots \text{[0.5 point]}
\end{aligned}$$

5 Short Answers

The following questions require a true/false accompanied by one sentence of explanation, or a reasonably short answer (usually at most a few sentences).

To discourage random guessing, no credit will be given for answers without a correct explanation.

(a) [2 points] [True / False] When dealing with high dimensional data, kNN model always produces better performance when you increase the value of K.

ANSWER: False. In the extreme case when K=N, the model becomes majority voting which is a baseline model that is easy to be outperformed.

(b) [2 points] [True / False] In logistic regression, sometime the gradient descent algorithm will converge to a local minimum and fail to find the global minimum even if you run the algorithm for a very long time (i.e. with unlimited time budget). This is why we prefer more advanced optimization algorithm such as conjugate gradient.

ANSWER: False. Logistic Regression is a convex optimization problem (with negative log likelihood as its objective function) which has a unique global minimum. Therefore gradient descent will cover as long as an appropriate learning rate is used and the algorithm is run for long enough. We use advanced optimization algorithm because they are faster and some of them do not require a pre-set learning rate. *KEY: convex/concave optimization, unique global minimum/maximum*

(c) [2 points] [True / False] PCA can both reduce and increase the dimension though increasing the dimension of input space using PCA will not lead to any gain of performance.

ANSWER: False. PCA can only reduce dimension but not increase it.

(d) [2 points][True / False] Increasing the number of layers in a deep neural network will increase the learning capability of the architecture and consequently decrease the test error. Therefore, large number of hidden layers is always preferred.

ANSWER: False. Larger number of hidden layer in principle means larger number of parameters which can lead to overfitting. *KEY: overfitting*

(e) [5 points] Suppose you are using logistic regression (trained with Maximum Likelihood Estimation), and the classifier is performing poorly (has unacceptably high error) on the test set but performing well on the training set. Determine whether each of the following is a reasonable attempt to get it up to an acceptable level of performance? (Yes/No) In either case, give a one-sentence justification.
Note: by the symptom described in the question, the model has the problem of overfitting.

i [1point] Increase the value of the regularization parameter.
ANSWER: Yes. The regularization parameter decides how much penalty is applied to the weight vector which represent the complexity of the model. With less complexity the model is less likely to overfit.

ii [1 point] Decrease the value of the regularization parameter.
ANSWER: No. The reason is explained in part i.

iii [1 point] Use Newton's method instead of gradient descent.
ANSWER: No. Logistic Regression has a convex objective function so gradient descent can perform well as long as the hyper parameters are carefully selected. Also the fact that your training error is low means that the optimization (learning parameters from the training data) is working fine. Using Newton's Method will not get much advantage on accuracy. *KEY: convex/concave*

iv [2 points] Get more training examples. In this case, please explain (1) whether the bias will increase or decrease; and (2) whether the variance will increase or decrease. (For simplicity, we assume that both the training and testing examples were randomly sampled from an IID distribution.)
ANSWER: Yes. More training data can help from overfitting. The bias will increase because it is more difficult to fit the training data when there are larger number of samples in the training set. However using more training data helps the model to generalize better so the variation will decrease.

(f) [2 points] Comparing regular Auto-encoder and Denoising Auto-encoder, in principle which one has higher bias and which one has higher variance? Please briefly explain your answer.
ANSWER: Denoising Auto-encoder will result in high bias compared to regular Auto-encoder since during training phase some corruption is manually injected to the training data. However this artificial noise helps the model to generalize better which results in lower variance. *KEY: generalization, regularization*

(g) [2 points] PCA is often used to preprocess the input data and reduce the dimension of the input space. In principle, will this process result in higher or lower bias? Please justify your answer with a one-sentence explanation.
ANSWER: In general, bias will increase after dimension reduction since reducing the dimensionality of the data will loss information which prevent the model from fitting to the training data as well as it would have done without using dimension reduction.

(h) [2 points] [True / False] In general SVM solves binary class classification problem where Gaussian Discriminant Analysis can also be applied. Any decision boundary found by a GDA model can in principle be reproduced by a SVM with polynomial kernel of degree of two.
ANSWER: True. In the scheme of binary class classification, a GDA model with shared covariance will produce a linear decision boundary while a GDA model with individual/different covariance for each class will produce a 2nd order polynomial decision boundary. Both of them can be produced by a SVM with polynomial kernel of degree of two. *Trick: can be $\neq$ will always be*

(i) [3 points] Consider applying Gaussian Discriminant Analysis (GDA) to a classification problem. Suppose the modeling assumptions made by GDA are true, and we have infinite training data. "Then, the learned GDA classifier will have zero training error". Is the statement true or false?
ANSWER: False. Every Gaussian model in the GDA model has non-zero probability cross the entire sample space which leaves a non-zero probability of misclassification at any point.

(j) [4 points] Explain what are the advantages and disadvantages of kernelization.

[2 poinits] Advantage(s):
ANSWER: map input data to more separable feature space; operate in high dimensional feature space; easy to switch among different kernel functions. Typically kernel function is designed so that it is much faster to compute than explicit inner products with feature vectors.

[2 points] Disadvantage(s):
ANSWER: When the number of training data is large, the computational cost can be prohibitively high.

(k) [4 points] Which of the following models are discriminative? Which ones are generative? Please briefly explain your answer. If the question is not clear or more information is needed then please elaborate the question.

(1) Logistic Regression
ANSWER: Discriminative. Logistic regression directly models the probability of $P(t|\mathbf{x};\theta)$, which is not modeling the data distribution.

(2) Gaussian Discriminant Analysis
ANSWER: Generative. GDA use Gaussian distributions to model how the data from each individual class is generated which is the probability of $P(\mathbf{x}|t = k)$ and prior $P(t = k)$ where k is the class label.

(3) Naive Bayes
ANSWER: Generative. NB model the probability of $P(t|\mathbf{x})$ by learning $P(\mathbf{x}|t)$ and $P(t)$ from the training data which makes it a generative model.

(4) Gaussian Mixture Model
ANSWER: Generative. GMM learns to represent the data by learning the probability of the data points $P(\mathbf{x})$ being generated by each individual Gaussian in the model.

(l) [2 points in total] Suppose that you want to solve a classification problem in a high-dimensional feature space (where there may be high correlations among the features). In principle, comparing logistic regression and Naive Bayes, which method is better suited to handle this problem? Please explain your answer.
ANSWER: Logistic regression is a better choice. When some of the features will be highly correlated, the assumption Naive Bayes makes, which is that features are mutually independent, is not valid. Consequently, Naive Bayes is not a desirable model for this situation. *KEY: NB assumption is no longer valid.*

(m) [4 points] Which of the following models produce a linear separating hyperplane in the feature space? Which ones produce nonlinear separating hyperplane in the feature space? Please briefly explain your answer. If the question is not clear or more information is needed, please elaborate the question.

(1) Logistic Regression
ANSWER: Linear. The decision boundary is eventually $\mathbf{w}^T\phi(\mathbf{x}) + b = 0$.

(2) Gaussian Discriminant Analysis with shared covariance
ANSWER: Linear. The second order terms are cancelled because the coefficients are both Σ^{-1}.

(3) Gaussian Discriminant Analysis with different covariances for each Gaussian
ANSWER: Non-linear. When the Gaussians use different covariance the second order terms do not cancel which makes the decision boundary(s) nonlinear.

(4) Neural Network
ANSWER: Both. A single layer NN produces linear decision boundary while a multilayer one with nonlinear activation function(s) produces nonlinear decision boundary. **Error Case:** it depends on if the activation function is linear or nonlinear.

(n) [3 points] Training with more training data usually improves generalization. Suppose we have a training set containing N training cases. If we duplicate each of the training cases L times so that the training set now contains LN cases (but still only N unique cases), will this improve the generalization performance of a logistic regression (in any case, assume that the regularization (e.g., $L2$) hyper-parameter will be carefully tuned on the validation data)?
ANSWER: No. Replicating the data is equivalent to training on the same training set for some more epochs. (i.e., objective function is scaled up by a constant factor N, assuming proper regularization after cross-validation).

(o) [2 points]Locally Weighted Linear regression model (as you've seen in Homework 1) has the error function of:

$$E(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{n} r^{(i)}(\mathbf{w}^T\phi(\mathbf{x}^{(i)}) - t^{(i)})^2$$

where

$$r^{(i)} = \exp\left(-\frac{\left(\phi(\mathbf{x}) - \phi(\mathbf{x}^{(i)})\right)^2}{2\tau^2}\right)$$

How will decreasing the value of τ affect the bias and variance of the model?
ANSWER: Smaller τ means the weight decreases faster when the distance between a training point and the test point increases. Therefore with smaller τ the model focus more on the training points that are closer to the test points which means there are fewer of them. This is similar to decreasing the number of training points which will result in lower bias but higher variance.

(p) [3 points] Below are four duplicated plots of a binary class classification problem's training data. Assuming SVMs with different kernels are applied to this dataset. Please draw a possible decision boundary in the case of applying each of these kernels: linear, 2nd order polynomial, RBF.
ANSWER: Below are plots made by Matlab svmtrain function. Any reasonable drawing will get full credit.

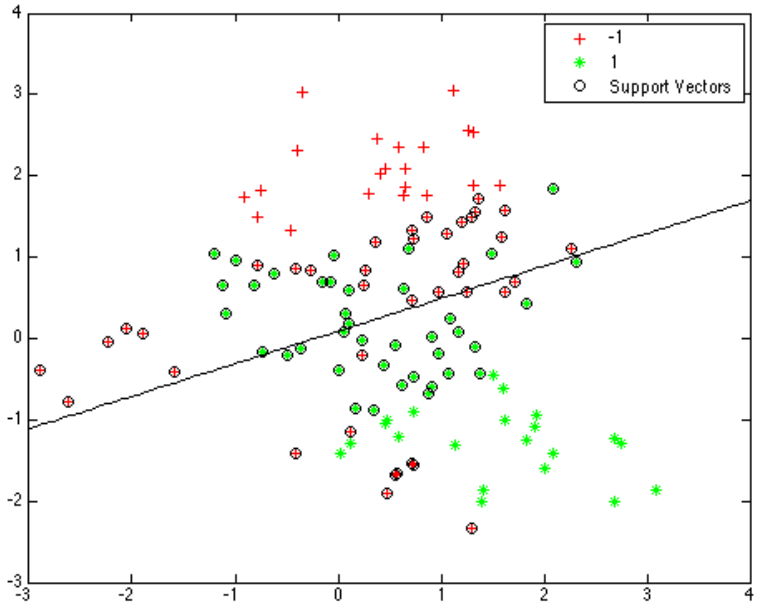

Figure 1: linear kernel

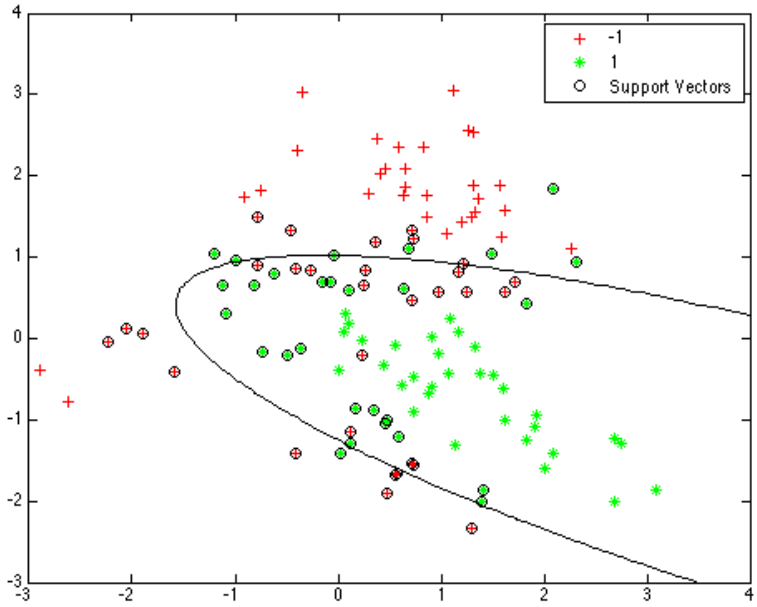

Figure 2: 2nd order polynomial kernel

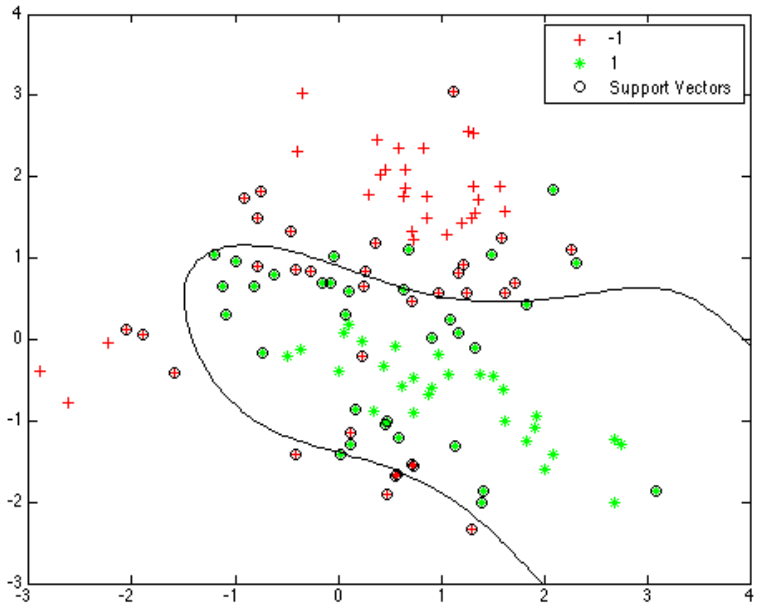

Figure 3: RBF kernel

(q) [4 points] Assuming that you have the 2D dataset as depicted by the duplicated figures below. In the first figure, draw the basis vectors of the PCA, label them by $\mathbf{b}_1, \mathbf{b}_2$ where $\mathbf{b}_1$ corresponds to the larger eigenvalue; in the second figure, draw the two bases that Sparse Coding will most likely learn. Hint: Sparse Coding learns to represent data by assuming $\mathbf{x}^{(i)} = \sum_{j=1}^{K} a_j^{(i)} \mathbf{b}_j$ where K is the number of basis vectors (i.e., in this problem, $K = 2$), $\mathbf{b}_j$ is called a basis vector and $a^{(i)}$ is a sparse vector called the "sparse code" (or encoding) for $\mathbf{x}^{(i)}$.

ANSWER: For Sparse Coding the order does not matter.

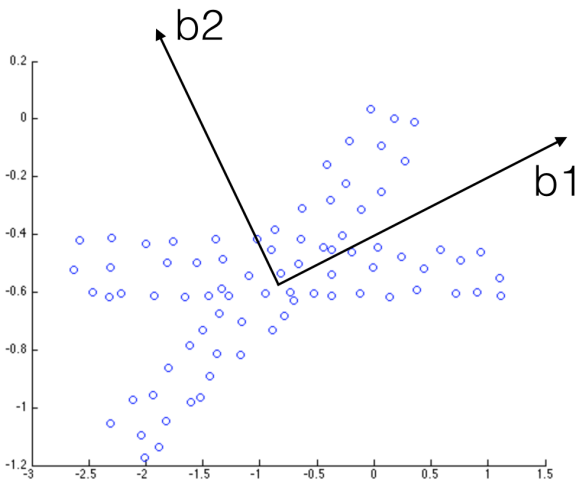

Figure 4: 2 basis found by PCA

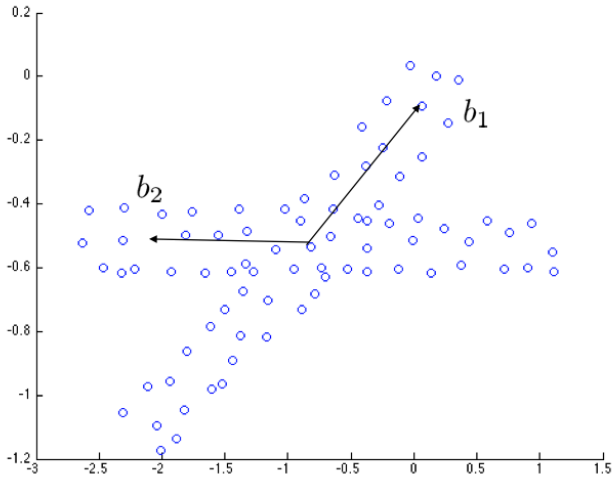

Figure 5: Two bases find by Sparse Coding

(This page is left blank intentionally. Please feel free to use this page if you need extra space for scratching. You may detach this page if you don't need it).

(This page is left blank intentionally. Please feel free to use this page if you need extra space for scratching. You may detach this page if you don't need it).