EECS545 Machine Learning
Solution for Homework #3

1 [10 points] MAP estimates and weight decay

By contradiction, assume that

$$\|\mathbf{w}_{MAP}\|_2 > \|\mathbf{w}_{ML}\|$$

Then, we have that

$$\begin{aligned} p(\mathbf{w} = \mathbf{w}_{MAP}) &= \frac{1}{(2\pi)^{\frac{M}{2}}|\sigma^2 I|^{\frac{1}{2}}} e^{(-\frac{1}{2\tau^2}\|\mathbf{w}_{MAP}\|_2^2)} \\ &< \frac{1}{(2\pi)^{\frac{M}{2}}\sigma^M} e^{(-\frac{1}{2\tau^2}\|\mathbf{w}_{ML}\|_2^2)} \\ &= p(\mathbf{w} = \mathbf{w}_{ML}) \end{aligned}$$

This yields

$$\begin{aligned} p(\mathbf{w} = \mathbf{w}_{MAP}) \prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{w} = \mathbf{w}_{MAP}) &< p(\mathbf{w} = \mathbf{w}_{ML}) \prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{w} = \mathbf{w}_{MAP}) \\ &\leq p(\mathbf{w} = \mathbf{w}_{ML}) \prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}, \mathbf{w} = \mathbf{w}_{ML}) \end{aligned}$$

where the last inequality holds since $\mathbf{w}_{ML}$ was chosen to maximize $\prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w})$. However, this result gives us a contradiction, since $\mathbf{w}_{MAP}$ was chosen to maximize $p(\mathbf{w}) \cdot \prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w})$.

Note: It is not necessary that the proof involves a contradiction argument.

2 [25 + 5 points] Direct construction of valid kernels

Notes:

- We will use G_i to denote the (Gram) matrix corresponding to the kernel function $k_i(.., .)$
- For the ones that are not kernels, it is sufficient to come up with one counter example (we only list one but there could be others). In addition, it is possible to prove of validity of all kernels by proving $\forall \mathbf{u}, \mathbf{u}^T G \mathbf{u} \geq 0$ by expanding $\mathbf{u}^T G \mathbf{u} = \sum_{i=1}^{N} \sum_{j=1}^{N} u_i G_{ik} u_k = \sum_{i=1}^{N} \sum_{j=1}^{N} u_i k\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) u_k \geq 0$. However, it is possible to take shortcuts by knowing some properties of Positive Semidefinite (PSD) Matrices. In particular, PSD + PSD = PSD, and, PSD * non-negative real number = PSD.
- All parts are worth 3 points each, except for part (e), which is worth 4 points.

(a) [2 points] Kernel. The sum of 2 positive semidefinite matrices is a positive semidefinite matrix: $\forall \mathbf{u}, \mathbf{u}^T G_1 \mathbf{u} \geq 0, \mathbf{u}^T G_2 \mathbf{u} \geq 0$ since k_1, k_2 are kernels. This implies $\forall \mathbf{u}, \mathbf{u}^T G \mathbf{u} = \mathbf{u}^T G_1 \mathbf{u} + \mathbf{u}^T G_2 \mathbf{u} \geq 0$

(b) [2 points] Not a kernel. Counterexample: let $k_2(\cdot,\cdot) = 2k_1(\cdot,\cdot)$ (we are using (1c) here to claim $2k_1$ is a kernel). Then we have $\forall \mathbf{u}, \mathbf{u}^T G \mathbf{u} = \mathbf{u}^T (G_1 - 2G_1)\mathbf{u} = -\mathbf{u}^T G_1 \mathbf{u} \leq 0$

(c) [2 points] Kernel. Multiplying a PSD by a non-negative integer produces a PSD. $\forall \mathbf{u}, \mathbf{u}^T G_1 \mathbf{u} \geq 0$, which implies $\forall \mathbf{u}, \quad a\mathbf{u}^T G_1 \mathbf{u} \geq 0$

(d) [2 points] Not a kernel. Counterexample: $a = 1$. Then we have $\forall \mathbf{u}, -\mathbf{u}^T G_1 \mathbf{u} \leq 0$

(e) [5 points] Kernel. k_1 is a kernel, thus $\exists \phi^{(1)}, k_1(\mathbf{x}, \mathbf{z}) = \phi^{(1)}(\mathbf{x})^T \phi^{(1)}(\mathbf{z}) = \sum_i \phi_i^{(1)}(\mathbf{x}) \phi_i^{(1)}(\mathbf{z})$. Similarly, k_2 is a kernel, thus $\exists \phi^{(2)}, k_2(\mathbf{x}, \mathbf{z}) = \phi^{(2)}(\mathbf{x})^T \phi^{(2)}(\mathbf{z}) = \sum_i \phi_i^{(2)}(\mathbf{x}) \phi_i^{(2)}(\mathbf{z})$

$$
\begin{aligned}
k(\mathrm{x}, \mathrm{z}) &= k_1(\mathrm{x}, \mathrm{z}) k_2(\mathrm{x}, \mathrm{z}) \\
&= \sum_i \phi_i^{(1)}(\mathrm{x}) \phi_i^{(1)}(\mathrm{z}) \sum_j \phi_j^{(2)}(\mathrm{x}) \phi_j^{(2)}(\mathrm{z}) \\
&= \sum_i \sum_j \phi_i^{(1)}(\mathrm{x}) \phi_i^{(1)}(\mathrm{z}) \phi_j^{(2)}(\mathrm{x}) \phi_j^{(2)}(\mathrm{z}) \\
&= \sum_i \sum_j \left(\phi_i^{(1)}(\mathrm{x}) \phi_j^{(2)}(\mathrm{x}) \right) \left(\phi_i^{(1)}(\mathrm{z}) \phi_j^{(2)}(\mathrm{z}) \right) \\
&= \sum_{(i,j)} \psi_{i,j}(\mathrm{x}) \psi_{i,j}(\mathrm{z})
\end{aligned}
$$

Where the last equality holds because we can define $\psi_{i,j}(\mathbf{x}) = \phi_i^{(1)}(\mathbf{x}) \phi_j^{(2)}(\mathbf{x})$. Therefore, we can define $\psi(\mathbf{x})$ to be a vector containing all $\psi_{i,j}(\mathbf{x})$ for all possible pairs (i, j). And we can rewrite:

$$k(\mathbf{x}, \mathbf{z}) = \psi(\mathbf{x})^T \psi(\mathbf{z})$$

since k can be written in this form, it is a kernel.

(f) [3 points] Kernel. Just let $\psi(\mathrm{x}) = f(\mathrm{x})$, and since $f(\mathrm{x})$ is a scalar, we have $k(\mathrm{x}, \mathrm{z}) = \psi(\mathrm{x})^T \psi(\mathrm{z})$ and we are done.

(g) [3 points] Kernel. since k_3 is a kernel, the matrix G_3 obtained for any finite set $\left\{\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(N)}\right\}$ is positive semidefinite, and so it is also positive semidefinite for the sets $\left\{\phi\left(\mathbf{x}^{(1)}\right), \ldots, \phi\left(\mathbf{x}^{(N)}\right)\right\}^2$

Alternatively, let $k_3(\mathbf{x}, \mathbf{z}) = \psi(\mathbf{x})^T \psi(\mathbf{z})$ so that $k(\mathbf{x}, \mathbf{z}) = \psi(\phi(\mathbf{x}))^T \psi(\phi(\mathbf{z})) = ((\psi \odot \phi)(\mathbf{x}))^T ((\psi \odot \phi)(\mathbf{z}))$

(h) [3 points] Kernel. By combining (1a) sum, (1c) scalar product, (1e) powers, (1f) constant term, we see that any polynomial of a kernel k_1 will again be a kernel.

(i) [3 points] $\phi(\mathbf{x}) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2} x_1 x_2 \\ \sqrt{2} x_1 \\ \sqrt{2} x_2 \\ 1 \end{bmatrix}$

(j) [5 points extra credit]

$$
\begin{aligned}
k(\mathrm{x},\mathrm{z}) &= \exp\left(-\frac{\|\mathrm{x}-\mathrm{z}\|^2}{2\sigma^2}\right) \\
&= \exp\left(-\frac{\|\mathrm{x}\|^2}{2\sigma^2}\right)\cdot\exp\left(-\frac{\|\mathrm{z}\|^2}{2\sigma^2}\right)\cdot\exp\left(\frac{\mathrm{x}^T\mathrm{z}}{\sigma^2}\right) \\
&= \exp\left(-\frac{\|\mathrm{x}\|^2}{2\sigma^2}\right)\cdot\exp\left(-\frac{\|\mathrm{z}\|^2}{2\sigma^2}\right)\cdot\sum_{n=0}^{\infty}\frac{1}{n!}\left(\frac{\mathrm{x}^T\mathrm{z}}{\sigma^2}\right)^n
\end{aligned}
$$

where

$$
\begin{aligned}
\frac{\left(\mathbf{x}^T\mathbf{z}\right)^n}{n!\sigma^{2n}} &= \frac{1}{n!\sigma^{2n}}\sum_{k_1+\cdots+k_d=n}\binom{n}{k_1,\cdots,k_d}\Pi_{i=1}^{d}\left(x_i z_i\right)^{k_i} \\
&= \left(\frac{\sqrt{\binom{n}{k_1^{(1)},\cdots,k_d^{(1)}}}}{\sigma^n\sqrt{n!}}\prod_{i=1}^{d}x_i^{k_i^{(1)}},\cdots,\frac{\sqrt{\binom{n}{k_1^{(m_n)},\cdots,k_d^{(m_n)}}}}{\sigma^n\sqrt{n!}}\prod_{i=1}^{d}x_i^{k_i^{(m_n)}}\right) \\
&\cdot\left(\frac{\sqrt{\binom{n}{k_1^{(1)},\cdots,k_d^{(1)}}}}{\sigma^n\sqrt{n!}}\prod_{i=1}^{d}z_i^{k_i^{(1)}},\cdots,\frac{\sqrt{\binom{n}{k_1^{(m_n)},\cdots,k_d^{(m_n)}}}}{\sigma^n\sqrt{n!}}\prod_{i=1}^{d}z_i^{k_i^{(m_n)}}\right) \\
&= \tilde{\phi}_n(\mathbf{x})^T\tilde{\phi}_n(\mathbf{z}).
\end{aligned}
$$

And we have

$$
\begin{aligned}
k(\mathbf{x},\mathbf{z}) &= \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right)\cdot\exp\left(-\frac{\|\mathbf{z}\|^2}{2\sigma^2}\right)\cdot\sum_{n=0}^{\infty}\frac{1}{n!}\left(\frac{\mathbf{x}^T\mathbf{z}}{\sigma^2}\right)^n \\
&= \left(\exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right)\sum_{n=0}^{\infty}\tilde{\phi}_n(\mathbf{x})\right)^T\left(\exp\left(-\frac{\|\mathbf{z}\|^2}{2\sigma^2}\right)\sum_{n=0}^{\infty}\tilde{\phi}_n(\mathbf{z})\right) \\
&= \phi(\mathbf{x})^T\phi(\mathbf{z})
\end{aligned}
$$

Note: Applying (a), (c), (e), (h) infinite times is NOT a valid proof.

3 [20 points] Kernelizing the Perceptron

Algorithm 1: Perceptron training algorithm

$\boldsymbol{\alpha}_0 \leftarrow \mathbf{0}$;
for $t = 0$ *to* $T-1$ **do**
  Pick a random training example $(\mathbf{x}^{(n)}, y^{(n)})$ from $\mathcal{D}$ (with replacement)
  $h \leftarrow \boldsymbol{\alpha}_t^T\mathbf{k}(\mathbf{x}^{(n)})$ where $\mathbf{k}(\mathbf{x}) = [\mathbf{k}(\mathbf{x},\mathbf{x}^{(1)}),\ldots,k(\mathbf{x},\mathbf{x}^{(N)})]^T$
  if $y^{(n)}h < 0$ **then**
    $\boldsymbol{\alpha}_{t+1} \leftarrow \boldsymbol{\alpha}_t + y^{(n)}\boldsymbol{e}^n$ where $\boldsymbol{e}^n$ is a one-hot vector with 1 at index n.
  end
end
return $\boldsymbol{\alpha}_T$

(a) Observe that

(i)

$$\begin{aligned}\mathbf{w}_{t+1} &= \mathbf{w}_t + \mathbf{I}\{y^{(n)}h(\phi(\mathbf{x}^{(n)})) < 0\} \cdot y^{(n)}\phi(\mathbf{x}^{(n)}) \\ &= \mathbf{\Phi}^T\boldsymbol{\alpha}_t + \mathbf{I}\{y^{(n)}h(\phi(\mathbf{x}^{(n)})) < 0\} \cdot y^{(n)}\phi(\mathbf{x}^{(n)}) \\ &= \mathbf{\Phi}^T \underbrace{\left(\boldsymbol{\alpha}_t + \mathbf{I}\{y^{(n)}h(\phi(\mathbf{x}^{(n)})) < 0\} \cdot y^{(n)}\boldsymbol{e}^n\right)}_{\boldsymbol{\alpha}_{t+1}}\end{aligned}$$

where, $\boldsymbol{e}^n$ is a one hot vector i.e., $e^n_n = 1$ and rest of the elements are zero (the n^{th} standard basis vector in $\mathbb{R}^N$).

It's not necessary to write the expression in terms of the one-hot vector and the indicator function, but the solution needs to have both cases [2 points for < 0, 1 point for ≥ 0].

(ii) [1.5 points] **Base case:** $\mathbf{w}_0 = \mathbf{0} = \mathbf{\Phi}^T\mathbf{0}$. Thus, $\boldsymbol{\alpha}_0 = \mathbf{0}$.

[1.5 point] **Induction step:** In part (i), we showed that $\mathbf{w}_t = \Phi^T\boldsymbol{\alpha}_t \implies \mathbf{w}_{t+1} = \Phi^T\boldsymbol{\alpha}_{t+1}$. Thus, we can conclude that for all $0 \leq t \leq T$, $\mathbf{w}_t = \Phi^T\boldsymbol{\alpha}_t$.

It's not necessary to prove this by induction. But, it's important to show that $\alpha_0 = \mathbf{0}$.

(b) (i) From part (a) (i), $\boldsymbol{\alpha}_{t+1} = \boldsymbol{\alpha}_t + \mathbf{I}\{y^{(n)}h(\phi(\mathbf{x}^{(n)})) < 0\} \cdot y^{(n)}\boldsymbol{e}^n = \begin{cases}\boldsymbol{\alpha}_t & y^{(n)}h(\mathbf{x}^n) \geq 0 \\ \boldsymbol{\alpha}_t + y^{(n)}\boldsymbol{e}^n & y^{(n)}h(\mathbf{x}^{(n)}) < 0\end{cases}$.

[2 points for when $\boldsymbol{\alpha}$ gets updated, 1 point for the other case].

[1 point] At most one entry of α_t needs to be updated to obtain α_{t+1}.

(ii)

$$\begin{aligned}h(\phi(\mathbf{x}^{(n)}), \mathbf{w}_t) &= \mathbf{w}_t^T\phi(\mathbf{x}^{(n)}) \\ &= \left(\mathbf{\Phi}^T\boldsymbol{\alpha}_t\right)^T\phi(\mathbf{x}^{(n)}) && \text{[Proved before, 1 point]} \\ &= \boldsymbol{\alpha}_t^T\mathbf{\Phi}\phi(\mathbf{x}^{(n)}) \\ &= \boldsymbol{\alpha}_t^T\begin{bmatrix}\phi(\mathbf{x}^{(1)})^T\phi(\mathbf{x}^{(n)}) \\ \vdots \\ \phi(\mathbf{x}^{(N)})^T\phi(\mathbf{x}^{(n)})\end{bmatrix} && \text{[1 points]} \\ &= \boldsymbol{\alpha}_t^T\begin{bmatrix}\mathbf{k}(\mathbf{x}^{(1)}, \mathbf{x}^{(n)}) \\ \vdots \\ \mathbf{k}(\mathbf{x}^{(N)}, \mathbf{x}^{(n)})\end{bmatrix} && \text{[2 points]} \\ &= \boldsymbol{\alpha}_t^T\mathbf{k}(\mathbf{x}^{(n)})\end{aligned}$$

where $\mathbf{k}(\mathbf{x}) = [\mathbf{k}(\mathbf{x}, \mathbf{x}^{(1)}), \ldots, k(\mathbf{x}, \mathbf{x}^{(N)})]^T$.

(c) [4 points] See Algorithm 1.

[Line 3 shouldn't have $\phi(\mathbf{x}^{(n)})$ or $\mathcal{D}_\phi$]

[All correct: 4 points, ϕ or $\mathbf{\Phi}$ present at one place: 2 points, more than one place: 0 points]

[2 points, needs to be written in kernel form] Output/decision of classifier is $\text{sign}(\boldsymbol{\alpha}_T^T\mathbf{k}(\mathbf{x}))$.

4 [25 points] Implementing Soft Margin SVM by Optimizing Primal Objective

(a) [5 points]

Let $J(\mathbf{w}, b, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$.

Let

$$\begin{aligned} \xi^{\mathbf{w},b} &= \arg\min_{\xi} J(\mathbf{w}, b, \xi) \\ &\text{s.t. } \xi_i \geq 0\ ,\ y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - \xi_i \quad \forall i \end{aligned} \tag{1}$$

We prove by contradiction that $\xi_i^{\mathbf{w},b} = \max\left(0, 1 - y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b)\right)$.

Assume $\zeta^{\mathbf{w},b}$ is optimal instead of $\xi^*_{\mathbf{w},b}$. This implies $\zeta_i^{\mathbf{w},b} \neq \xi_i^{\mathbf{w},b}$ for some i.

Since $\zeta_i^{\mathbf{w},b}$ is required to satisfy the constraints in optimization problem (1),

$\zeta_i^{\mathbf{w},b} \geq \max\left(0, 1 - y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b)\right) = \xi_i^{\mathbf{w},b}$.

Hence, $\zeta_i^{\mathbf{w},b} > \xi_i^{\mathbf{w},b}$ as a result of our assumption. This implies that $J(\mathbf{w}, b, \zeta^{\mathbf{w},b}) > J(\mathbf{w}, b, \xi^{\mathbf{w},b})$ which leads to a contradiction.

Thus,

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} J(\mathbf{w}, b, \xi) \qquad \text{s.t.} \quad \xi_i \geq 0\ ,\ y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - \xi_i \quad \forall i \\ \equiv &\min_{\mathbf{w},b}\left(\min_{\xi} J(\mathbf{w}, b, \xi) \qquad \text{s.t.} \quad \xi_i \geq 0\ ,\ y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - \xi_i \quad \forall i\right) \\ \equiv &\min_{\mathbf{w},b} J(\mathbf{w}, b, \xi^{\mathbf{w},b}) \\ \equiv &\min_{\mathbf{w},b} E(\mathbf{w}, b) \end{aligned}$$

Note: Such a detailed proof is not expected. As long as the solution shows that $\xi_i \geq \max\left(0, 1 - y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b)\right)$ and then claims that equality holds since J wants to minimize ξ_i, full points will be awarded.

(b) [6 points] The max(0,) operator can be thought of a piece-wise function. Calculating the derivative of a piece-wise function can be done by taking the derivative of each piece. In particular, let $f = \max(0, k)$ then $\frac{\partial f}{\partial z} = \frac{\partial k}{\partial z}$ if $k > 0$, and $\frac{\partial f}{\partial z} = \frac{\partial 0}{\partial z} = 0$ if $0 > k$. We can compactly write $\frac{\partial}{\partial z}\max(0, k) = \mathbf{I}[k > 0]\frac{\partial k}{\partial z}$ Alternatively (which we will use below), we can rewrite $\max(0,\ k) = \mathbf{I}[k > 0]k$ The error function we are trying to minimize is:

$$\begin{aligned} E(\mathbf{w}, b) &= \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\max\left(0, 1 - y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right)\right) \\ &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\mathbf{I}\left[1 - y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right) > 0\right]\left(1 - y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right)\right) \\ &= \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\mathbf{I}\left[y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right) < 1\right]\left(1 - y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right)\right) \end{aligned}$$

Then, it easily follows that:

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

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

Note: Skipping information about what to do with max(.) loses 2 points.

(c) [9 points] Both cases are the correct answers.
Case 1. $\alpha_0 = \eta_0$ Parameters and accuracies:

- Iter 5: accuracy = 54.17%
  weight=[112.000, -42.750, 272.500, 103.000]
  bias=-0.124
- Iter 50: accuracy = 95.83%
  weight=[-2.020, -11.941, 25.853, 11.549]
  bias=-0.373
- Iter 100: accuracy = 95.83%
  weight=[-2.559, -5.282, 11.376, 5.757]
  bias=-0.383
- Iter 1000: accuracy = 95.83%
  weight=[-0.464, -0.326, 1.054, 1.279]
  bias=-0.404
- Iter 5000: accuracy = 95.83%
  weight=[-0.321, -0.279, 0.893, 0.987]
  bias=-0.418
- Iter 6000: accuracy = 95.83%
  weight=[-0.329, -0.282, 0.886, 0.975]
  bias=-0.420

Case 2. $\alpha_0 = \eta_0/(1 + \eta_0)$ Parameters and accuracies:

- Iter 5: accuracy = 54.17%
  weight=[96.000, -36.643, 233.571, 88.286]
  bias=-0.069
- Iter 50: accuracy = 95.83%
  weight=[-1.981, -11.712, 25.356, 11.327]
  bias=-0.287
- Iter 100: accuracy = 95.83%
  weight=[-1.990, -4.819, 11.451, 5.740]
  bias=-0.296
- Iter 1000: accuracy = 95.83%
  weight=[-0.500, -0.324, 1.055, 1.283]
  bias=-0.318
- Iter 5000: accuracy = 95.83%
  weight=[-0.352, -0.278, 0.886, 1.003]
  bias=-0.333

- Iter 6000: accuracy = 95.83%
  weight=[-0.337, -0.281, 0.894, 0.986]
  bias=-0.334

Note: It is ok if your b and w were completely different for smaller iterations, though they should be within $\pm 3\%$ range to the answer for 6000 iterations.

(d) [4 points.] 2 points each:

$$\nabla_{\mathbf{w}} E^{(i)}(\mathbf{w}, b) = \frac{1}{N}\mathbf{w} - C\mathbf{I}\left[y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right) < 1\right] y^{(i)}\mathbf{x}^{(i)} \tag{2}$$

$$\frac{\partial}{\partial b} E^{(i)}(\mathbf{w}, b) = -C\mathbf{I}\left[y^{(i)}\left(\mathbf{w}^T\mathbf{x}^{(i)} + b\right) < 1\right] y^{(i)} \tag{3}$$

(e) [6 points] Both cases are the correct answers.
Case 1. $\alpha_0 = \eta_0$ Parameters and accuracies:

- Iter 5: accuracy = 95.83%
  weight=[-1.781, -3.128, 8.554, 5.203]
  bias=-0.054
- Iter 50: accuracy = 95.83%
  weight=[-1.379, 0.001, 2.587, 2.856]
  bias=-0.087
- Iter 100: accuracy = 95.83%
  weight=[-1.257, 0.114, 1.709, 2.317]
  bias=-0.094
- Iter 1000: accuracy = 95.83%
  weight=[-0.489, -0.190, 0.957, 1.140]
  bias=-0.120
- Iter 5000: accuracy = 95.83%
  weight=[-0.428, -0.235, 0.889, 1.065]
  bias=-0.139
- Iter 6000: accuracy = 95.83%
  weight=[-0.442, -0.214, 0.910, 1.064]
  bias=-0.140

Case 2. $\alpha_0 = \eta_0/(1 + \eta_0)$ Parameters and accuracies:

- Iter 5: accuracy = 95.83%
  weight=[-1.605, -2.830, 7.755, 4.700]
  bias=-0.039
- Iter 50: accuracy = 95.83%
  weight=[-1.689, -0.180, 2.503, 2.783]
  bias=-0.071
- Iter 100: accuracy = 95.83%
  weight=[-1.213, 0.086, 1.681, 2.202]
  bias=-0.077
- Iter 1000: accuracy = 95.83%
  weight=[-0.495, -0.189, 0.954, 1.149]
  bias=-0.103

- Iter 5000: accuracy = 95.83%
  weight=[-0.424, -0.238, 0.889, 1.062]
  bias=-0.122
- Iter 6000: accuracy = 95.83%
  weight=[-0.443, -0.217, 0.907, 1.063]
  bias=-0.123

Note: It is ok if your b and w were completely different for smaller iterations, though they should be within $\pm 3\%$ range to the answer for 6000 iterations.

5 [15 points] Scikit-learn SVM for classifying SPAM

(a) See the solution code q5.ipynb.

(b) The test error of SVM for different training set sizes:

- Training set size 50: Test set error = 5.000%
- Training set size 100: Test set error = 3.000%
- Training set size 200: Test set error = 1.250%
- Training set size 400: Test set error = 1.000%
- Training set size 800: Test set error = 1.000%
- Training set size 1400: Test set error = 0.8750%

The data set size 1400 gives the best performance.

(c) [3 points] The test error of Naive Bayes are:

- Training set size 50: Test set error = 3.875%
- Training set size 100: Test set error = 2.625%
- Training set size 200: Test set error = 2.625%
- Training set size 400: Test set error = 1.875%
- Training set size 800: Test set error = 1.75%
- Training set size 1400: Test set error = 1.625%

The test error of SVM is larger than that of Naive Bayes when training set size is small (SVM is likely to overfit because of the lack of training data). When the training set is large enough, SVM outperforms Naive Bayes.

Credits

Some questions adopted/adapted from http://www.stanford.edu/class/cs229/ and from Bishop PRML.