Homework 2 **Daniel Wivagg**
RBE 549 Due Date: 1/23/19
Prof. Jacob Whitehill

Problem 1

$$J(\theta) = \frac{1}{4}\sum_{x\in X}(f^*(x) - f(x;\theta))^2 \Rightarrow J(w,b) = \frac{1}{4}\sum_{x\in X}\left[f^*(x) - (x^\top w + b)\right]^2 \tag{6.1}$$

To find the values that minimize the loss function, the gradient of Eqn. 6.1 is calculated with respect to w and b as shown below.

$$\begin{aligned}\nabla_w J(w,b) &= \frac{1}{4}\sum_{x\in X}\nabla_w\left[f^*(x) - (x^\top w + b)\right]^2 \\ &= \frac{1}{2}\sum_{x\in X} xf^*(x) - xx^\top w - xb\end{aligned}$$

$$\begin{aligned}\nabla_b J(w,b) &= \frac{1}{4}\sum_{x\in X}\nabla_b\left[f^*(x) - (x^\top w + b)\right]^2 \\ &= \frac{1}{2}\sum_{x\in X} f^*(x) - x^\top w - b\end{aligned}$$

Now, the gradient with respect to w is set to zero in order to find the optimal solutions. The summations are evaluated, given we know the features and labels of each instance.

$$\begin{aligned}0 &= \frac{1}{2}\sum_{x\in X} xf^*(x) - xx^\top w - xb \\ &= \sum_{x\in X} xf^*(x) - \sum_{x\in X} xx^\top w - \sum_{x\in X} xb \\ &= \begin{bmatrix}1\\1\end{bmatrix} - \begin{bmatrix}2w_1 + w_2\\ w_1 + 2w_2\end{bmatrix} - \begin{bmatrix}2b\\2b\end{bmatrix}\end{aligned}$$

Here, there are 3 unknowns but only 2 equations, so we use the gradient with respect to b to obtain one more.

$$\begin{aligned}0 &= \frac{1}{2}\sum_{x\in X} f^*(x) - x^\top w - b \\ &= \sum_{x\in X} f^*(x) - \sum_{x\in X} x^\top w - \sum_{x\in X} b \\ &= 2 - (2w_1 + 2w_2) - 4b\end{aligned}$$

Combining all three equations into one linear system, we can find the final values for w and b.

$$\begin{bmatrix} 2 & 1 & 2 \\ 1 & 2 & 2 \\ 2 & 2 & 4 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ b \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} w_1 \\ w_2 \\ b \end{bmatrix} = \begin{bmatrix} 2 & 1 & 2 \\ 1 & 2 & 2 \\ 2 & 2 & 4 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$$

$$\begin{bmatrix} w_1 \\ w_2 \\ b \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\frac{1}{2} \\ 0 & 1 & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{2} & \frac{3}{4} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{2} \end{bmatrix}$$

Problem 2

Part A One-shot solution

Given the cost function:

$$J(w) = \frac{1}{2n} \sum_{j=1}^{m} (\hat{y}^{(j)} - y^{(j)})^2 + \frac{\alpha}{2} w^\top w = \frac{1}{2n} (X^\top w - y)^2 + \frac{\alpha}{2} w^\top w$$

The gradient can be computed:

$$\begin{aligned} \nabla_w J(w) &= \nabla_w \frac{1}{2n} (X^\top w - y)^2 + \frac{\alpha}{2} w^\top w \\ &= \frac{1}{n} X (X^\top w - y) + \alpha w \\ &= \frac{1}{n} X X^\top w - \frac{1}{n} X y + \alpha w \end{aligned}$$

Setting the gradient equal to zero, the solution can be found:

$$\begin{aligned} \frac{1}{n} X X^\top w - \frac{1}{n} X y + \alpha w &= 0 \\ X X^\top w + \alpha w &= X y \\ \left(X X^\top + \alpha I \right) w &= X y \\ w &= \left(X X^\top + \alpha I \right)^{-1} X y \end{aligned}$$

The n term can be dropped entirely since α is a tuned hyperparameter, and $n\alpha$ simply rescales α. This was translated to the Python function for Method 1, and the *unregularized* cost is reported below for two different values of α.

Unregularized Training Cost, $\alpha = 1000$: 0.0697

Unregularized Testing Cost, $\alpha = 1000$: 0.0814

Unregularized Training Cost, $\alpha = 0$: 0.0563

Unregularized Testing Cost, $\alpha = 0$: 0.1041

Compared to using $\alpha = 0$, the regularization parameter $\alpha = 1000$ results in a higher training cost but a lower testing cost. This is likely because the higher regularization parameter does not allow for overfitting of the training data.

Part B Gradient descent

The values for testing and training during gradient descent are relatively close to the one-shot solution with $\alpha = 1000$. However, in this case the regularization hyperparameter must be smaller to account for the small size of the mean-squared error term.

Unregularized Training Cost, $\alpha = 0.5 : 0.0943$

Unregularized Testing Cost, $\alpha = 0.5 : 0.0968$

Problem 3

For this problem, we will augment the existing features, labels, and weights to account for a bias term. Let $\tilde{w} = \begin{bmatrix} w \\ b \end{bmatrix}$ (a 577x1 column vector of weights and the bias term), $\tilde{X} = \begin{bmatrix} X \\ 1 \end{bmatrix}$ (a 577x2000 matrix of training instances), and $\tilde{y} = \begin{bmatrix} y \\ 0 \end{bmatrix}$ (a 2000x1 column vector of training labels). Also, the weights for each training example shall be denoted by the matrix $A = \begin{bmatrix} a_1 & \cdots & a_{2000} \end{bmatrix} I_{2000\times 2000}$ since these are constant predetermined values. Now, the updated cost function is as follows.

$$J(w,b) = \frac{1}{2n}(\tilde{X}^\top \tilde{w} - A\tilde{y})^\top (\tilde{X}^\top \tilde{w} - A\tilde{y}) + \frac{\alpha}{2}\tilde{w}^\top \begin{bmatrix} I_{576\times 576} & \vec{0} \\ \vec{0}^\top & 0 \end{bmatrix} \tilde{w}$$

Now, the gradient must be taken with respect to w and b separately.

$$\begin{aligned} \nabla_w J(w,b) &= \nabla_w \left[\frac{1}{2n}(\tilde{X}^\top \tilde{w} - A\tilde{y})^\top (\tilde{X}^\top \tilde{w} - A\tilde{y}) + \frac{\alpha}{2}\tilde{w}^\top \begin{bmatrix} I & \vec{0} \\ \vec{0}^\top & 0 \end{bmatrix} \tilde{w} \right] \\ &= \frac{1}{n} \begin{bmatrix} X \\ 0 \end{bmatrix} (\tilde{X}^\top \tilde{w} - A\tilde{y}) + \alpha \begin{bmatrix} w \\ 0 \end{bmatrix} \end{aligned}$$

$$\begin{aligned} \nabla_b J(w,b) &= \nabla_b \left[\frac{1}{2n}(\tilde{X}^\top \tilde{w} - A\tilde{y})^\top (\tilde{X}^\top \tilde{w} - A\tilde{y}) + \frac{\alpha}{2}\tilde{w}^\top \begin{bmatrix} I & \vec{0} \\ \vec{0}^\top & 0 \end{bmatrix} \tilde{w} \right] \\ &= \frac{1}{n} \begin{bmatrix} 0_X \\ 1 \end{bmatrix} (\tilde{X}^\top \tilde{w} - A\tilde{y}) \end{aligned}$$

Finally, the optimal solution can be found by setting both gradients equal to 0. This causes the n to disappear entirely, as before, since multiplying na simply results in a rescaling of a. For this portion, let $\tilde{X}_w = \begin{bmatrix} X \\ 0 \end{bmatrix}$, which is the gradient of $\tilde{X}$ with respect to w; $\tilde{X}_b = \begin{bmatrix} 0_X \\ 1 \end{bmatrix}$, which is the gradient of $\tilde{X}$ with respect to b; and let $I_\alpha = \begin{bmatrix} I & \vec{0} \\ \vec{0}^\top & 0 \end{bmatrix}$, which is simply the matrix from

above for distributing α over w and not b.

$$\begin{aligned}
\tilde{X}_w(\tilde{X}^\top \tilde{w} - A\tilde{y}) + \alpha I_\alpha \tilde{w} &= 0 \\
\tilde{X}_w \tilde{X}^\top \tilde{w} - \tilde{X}_w A\tilde{y} + \alpha I_\alpha \tilde{w} &= 0 \\
\tilde{X}_w \tilde{X}^\top \tilde{w} + \alpha I_\alpha \tilde{w} &= \tilde{X}_w A\tilde{y} \\
(\tilde{X}_w \tilde{X}^\top + \alpha I_\alpha)\tilde{w} &= \tilde{X}_w A\tilde{y} \\
\tilde{w} &= (\tilde{X}_w \tilde{X}^\top + \alpha I)^{-1} \tilde{X}_w A\tilde{y}
\end{aligned}$$

Because the last element of $\tilde{y}$ is 0, the last element of $\tilde{X}_w A\tilde{y}$ will also be 0, and subsequently the entire equation evaluates to leave $b = 0$. Thus, we have found all of the weights, but not the bias term. This can be found by optimizing with respect to b.

$$\begin{aligned}
\tilde{X}_b(\tilde{X}^\top \tilde{w} - A\tilde{y}) &= 0 \\
\tilde{X}_b \tilde{X}^\top \tilde{w} - \tilde{X}_b A\tilde{y} &= 0 \\
\tilde{X}_b \tilde{X}^\top \tilde{w} &= \tilde{X}_b A\tilde{y} \\
\tilde{w} &= (\tilde{X}_b \tilde{X}^\top)^{-1} \tilde{X}_b A\tilde{y}
\end{aligned}$$

This equation solves for the b element of $\tilde{w}$ only, since $\tilde{X}_b A\tilde{y}$ only has a non-zero value in the final element. Now, we have a closed-form solution for both the weights and the bias term, with weighted training examples.

Problem 4

Considering a 1×2 image, the a regularization matrix S that would discourage asymmetry is as follows:

$$S = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

For the two-pixel case, this works well because the regularization term ends up being larger the greater the asymmetry, or difference, between the pixels. If we expand the equation, it actually becomes the squared error between the two pixels.

$$\begin{aligned}
w^\top S w &= \begin{bmatrix} w_1 & w_2 \end{bmatrix} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \\
&= w_1(w_1 - w_2) + w_2(w_2 - w_1) \\
&= (w_1 - w_2)^2
\end{aligned}$$

Therefore, the greater the asymmetry, the greater the disparity between w_1 and w_2, so the regularization term will favor weights that are more symmetrical.

Problem 5

The teacher's estimate of the state given all past observations is given by $P(x_t|y_1, \ldots, y_t)$. By the derivation below using Bayes' rule and some laws of probability, this state estimate can be

updated recursively.

$$
\begin{aligned}
P(x_t|y_1,\ldots,y_t) &= \frac{P(y_t|x_t,y_1,\ldots,y_{t-1})P(x_t|y_1,\ldots,y_{t-1})}{P(y_t|y_1,\ldots,y_{t-1})} && (1)\\
&\propto P(y_t|x_t,y_1,\ldots,y_{t-1})P(x_t|y_1,\ldots,y_{t-1}) && (2)\\
&= P(y_t|x_t)P(x_t|y_1,\ldots,y_{t-1}) && (3)\\
&= \frac{P(y_t|x_t)P(x_t,y_1,\ldots,y_{t-1})}{P(y_1,\ldots,y_{t-1})} && (4)\\
&= P(y_t|x_t)\frac{\sum_{x_{t-1}} P(x_t,x_{t-1},y_1,\ldots,y_{t-1})}{P(y_1,\ldots,y_{t-1})} && (5)\\
&= P(y_t|x_t)\frac{\sum_{x_{t-1}} P(x_t|x_{t-1},y_1,\ldots,y_{t-1})P(x_{t-1}|y_1,\ldots,y_{t-1})P(y_1,\ldots,y_{t-1})}{P(y_1,\ldots,y_{t-1})} && (6)\\
&= P(y_t|x_t)\sum_{x_{t-1}} P(x_t|x_{t-1},y_1,\ldots,y_{t-1})P(x_{t-1}|y_1,\ldots,y_{t-1}) && (7)\\
&= P(y_t|x_t)\sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}|y_1,\ldots,y_{t-1}) && (8)
\end{aligned}
$$

This is the desired final form. Notes:

1. Bayes' rule is applied.
2. Denominator is not necessary because the result can be normalized using the law of total probability.
3. Behavior depends on current state only, simplify.
4. Law of conditional probability is applied.
5. Marginal probability rule is applied.
6. Law of conditional probability is applied using the chain rule.
7. Cancel the denominator.
8. State estimate depends only on the previous state, final desired form is obtained.