COMS 4771 Machine Learning (Spring 2018)
Problem Set #2

Zhuangyu Ren, Yuchen Mo, Chengzhi Mao - UNI: zr2209, ym2706, cm3797

March 27, 2019

Problem 1

Similar to the proof of SVM duality, let the Lagrangian be $f(x, \lambda) = ||x - x_a||^2 + \lambda(w \cdot x + w_0)$. The constraint is a constant so it must be linear (affine).

$$f'_x = 2(x - x_a) + \lambda w = 0$$
$$x = x_a - \frac{\lambda w}{2}$$

Bring this constraint back to our Lagrangian, we have $g(\lambda) = \frac{\lambda^2 ||w||^2}{4} + \lambda(w \cdot x_a - \frac{\lambda}{2}||w||^2 + w_0)$

$$\begin{aligned} g'_\lambda &= \frac{||w||^2 \lambda}{2} + w \cdot x_a - ||w||^2 \lambda + w_0 \\ &= -\frac{||w||^2}{2}\lambda + x_a \cdot w + w_0 = 0 \\ \lambda &= \frac{2(x_a \cdot w + w_0)}{||w||^2} = \frac{2g(x_a)}{||w||^2} \end{aligned}$$

Thus

$$||x - x_a|| = \sqrt{\frac{\lambda^2 ||w||^2}{4}} = \frac{|g(x_a)|}{||w||}$$

Problem 2

2.1

Dataset: all data with the margin γ to the decision boundary and all have length R.
Order of updates: update the data that is on the decision boundary each time.
So suppose the algorithm makes a mistake in iteration t, because every data point has a margin γ and $\|x\|^2 = R^2$, then:

$$\overrightarrow{w}^{(t)} \cdot \overrightarrow{w}^* = (\overrightarrow{w}^{(t-1)} + y\overrightarrow{x}) \cdot \overrightarrow{w}^* = \overrightarrow{w}^{(t-1)} + \gamma$$

$$\left\|\overrightarrow{w}^{(t)}\right\|^2 = \left\|\overrightarrow{w}^{(t-1)} + y\overrightarrow{x}\right\|^2 = \left\|\overrightarrow{w}^{(t-1)}\right\|^2 + R^2$$

So, after T rounds,

$$T\gamma = \overrightarrow{w}^{(T)} \cdot \overrightarrow{w}^* = \left\|\overrightarrow{w}^{(T)}\right\| \|\overrightarrow{w}^*\| = R\sqrt{T}$$

The bound of mistakes made by the perceptron algorithm is tight.

2.2

$$\begin{aligned}
\|w_T\|^2 &= \|w_{T-1} + y\overrightarrow{x}\|^2 \\
&= \|w_{T-1}\|^2 + 2y(w_{T-1} \cdot \overrightarrow{x}) + \|y\overrightarrow{x}\|^2 \\
&= \|w_{T-1}\|^2 + 2y(w_{T-1} \cdot \overrightarrow{x}) + \|\overrightarrow{x_{iT}}\|^2 \\
&\leqslant \|w_{T-1}\|^2 + \|\overrightarrow{x_{iT}}\|^2 \\
&\leqslant \sum_{t=1}^{T} \|\overrightarrow{x_{iT}}\|^2
\end{aligned}$$

$$\|(I-P)x_i\|^2 = \|x_i\|^2 - 2Px_i^2 + \|Px_i\|^2$$

Here, $P = w^* w^{*T}$

$$P^2 x_i^2 = (w^* w^{*T})(w^* w^{*T}) x_i^2 = w^*(w^{*T} w^*) w^{*T} x_i^2 = (w^* w^{*T}) x_i^2 = Px_i^2$$

So,

$$\|(I-P)x_i\|^2 = \|x_i\|^2 - \|Px_i\|^2 \leqslant \varepsilon^2$$
$$\|x_i\|^2 \leqslant \varepsilon^2 + \|Px_i\|^2$$

After T times of mistakes,

$$\|w_T\|^2 \leqslant \varepsilon^2 T + \sum_{t=1}^{T} \|Px_{i_t}\|^2$$

2.3

$$\begin{aligned} w_T \cdot w^* &= (w_{T-1} + y\overrightarrow{x}) \cdot w^* \\ &= w_{T-1} \cdot w^* + y_{i_T} x_{i_T} \cdot w^* \\ &= \sum_{t=1}^{T} y_{i_t} x_{i_t} \cdot w^* \end{aligned}$$

And,

$$\begin{aligned} (w_T \cdot w^*)^2 &= (\sum_{t=1}^{T} y_{i_t} x_{i_t} \cdot w^*)^2 \\ &= \sum_{i=t}^{T} (y_{i_t} x_{i_t} \cdot w^*)^2 + \sum_{i=1}^{T} \sum_{j=1, j\neq i}^{T} y_{i_t} (w^* \cdot x_{i_t}) y_{j_t} (w^* \cdot x_{j_t}) \\ &\geqslant \sum_{i=t}^{T} (y_{i_t} x_{i_t} \cdot w^*)^2 + T(T-1)\gamma^2 \end{aligned}$$

Here,

$$y_{i_t}^2 = 1$$

$$(x_{i_t} \cdot w^*)^2 = x_{i_t}^T w^* {w^*}^T x_{i_t} = x_{i_T}^T P x_{i_t} = \|P x_{i_t}\|^2$$

So,

$$(w_T \cdot w^*)^2 \geq T(T-1)\gamma^2 + \sum_{t=1}^{T} \|P x_{i_t}\|^2$$

2.4

$$T(T-1)\gamma^2 + \sum_{t=1}^{T} \|P x_{i_t}\|^2 \leqslant (w_T \cdot w^*)^2 \leqslant \|w_T\|^2 \cdot \|w^*\|^2 \leqslant \varepsilon^2 T + \sum_{t=1}^{T} \|P x_{i_t}\|^2$$

It means that,

$$T(T-1)\gamma^2 \leqslant \varepsilon^2 T$$

So,

$$T \leqslant (\frac{\varepsilon}{\gamma})^2 + 1$$

Problem 3

3.1

e.g. $\varphi(x) = (x_1 \wedge \bar{x_2}) \vee (\bar{x_1} \wedge x_2)$
S = {((0, 0), 0), ((0, 1), 1), ((1, 0), 1), ((1, 1), 0)}. This set S is obviously not linearly separable.

3.2

For arbitrary $a, b \in \{0,1\}^d$, consider these parts of $\phi(a)$ and $\phi(b)$.

- Dims of $\phi(a)$ consists of 0 a_i or $\bar{a_i}$ items, $i \in \{1, \ldots, d\}$. The only dim satisfying this is $\phi_0(a)$ and must be 1, and the same for b. This dim contribute 1 to the dot product.
- Dims of $\phi(a)$ consists of 1 a_i or $\bar{a_i}$ items, $i \in \{1, \ldots, d\}$. We have $2d$ dims satisfying this, and clearly their contribution to $\phi(a) \cdot \phi(b)$ is the number of the j's s.t. $a_j = b_j$, $j \in \{1, \ldots, d\}$
- Dims of $\phi(a)$ consists of k a_i or $\bar{a_i}$ items, $2 \leq k \leq d,\ i \in \{1, \ldots, d\}$. For a certain set of j's, i.e., $j_1, j_2, \ldots, j_k$, satisfying $1 \leq j_1 < j_2 < \cdots < j_k \leq d$, we have 2^k dims containing these k items, and this group will contribute 1 to the dot product if and only if $a_{j_t} = b_{j_t}$ is true for all $t = 1, \ldots, k$. Furthermore, we have $\binom{d}{k}$ such groups. We denote the group of j's satisfying $a_j = b_j$ as J, then those groups will contribute $\binom{||J||}{k}$ to the dot product.

To conclude, by dividing $\phi(a)$ and $\phi(b)$ into parts with different number of basic logic elements. Let $T = ||J||$

$$\begin{aligned}\phi(a) \cdot \phi(b) &= 1 + T + \binom{T}{2} + \binom{T}{3} + \cdots + \binom{T}{T} \\ &= 2^T\end{aligned}$$

Then the corresponding 'kernel trick' is to count how many j's satisfies $a_j = b_j,\ j = 1, \ldots, d$, which is obviously in $O(d)$ time complexity.

3.3

From the definition of w^*. All its dimensions except the first is either 0 or 1. Thus, from the properties of dot product we know that $w^* \cdot \phi(x)$ is the sum over those dims of $\phi(x)$ that corresponds to the original disjunctions in $\varphi(x)$. Therefore, let C_{all} be the number of conjunctions in $\phi(x)$ and C_1 be the number of conjunctions satisfied in $\phi(x)$

$$w^* \cdot \phi(x) = C_1 - 0.5$$

And

$$\frac{w^*}{||w^*||} \cdot \phi(x) = \frac{C_1 - 0.5}{\sqrt{C_{all} + 0.25}}$$

Let s be the number of conjunctions in φ. As we know $C_1 \leq C_{all}$ then the margin γ is upper bounded by $\frac{1}{\sqrt{4s+1}}$

3.4

As discussed in 3.2, now we consider $\phi(a) \cdot \phi(a)$. Then, the first dim contributed 1 to the dot product, those unary dims contributed no more than d, binary dims no more than $\binom{d}{2}$, etc. Therefore, given d, the radius of dataset S is upper bounded by $\max ||\phi(x^{(i)})|| = \sqrt{2^d} = 2^{d/2}$.

3.5

For the time complexity, as we need $O(d)$ time to compute $K(a, b)$ (as proved in 3.2), one iteration over the dataset takes $nO(d) = O(nd)$ time.
Suppose now we are making the t'th mistake. In the kernel space, we have

$$\begin{aligned} w^{(t)} \cdot w^* &= \left(w^{(t-1)} + y \cdot \phi(x)\right) \cdot w^* \\ &\geq w^{(t-1)} \cdot w^* + \gamma \\ &\geq \gamma t \end{aligned}$$

And

$$\begin{aligned} ||w^t||^2 &= ||w^{t-1} + y \cdot \phi(x)||^2 \\ &= ||w^{t-1}||^2 + 2yw^{t-1} \cdot \phi(x) + ||\phi(x)||^2 \\ &\leq ||w^{t-1}||^2 + R^2 \\ &\leq 2^d t \end{aligned}$$

Combine these two inequalities together we have

$$\begin{aligned} (\gamma T)^2 \leq (w^{(t)} \cdot w^*)^2 \leq ||w^*||^2 ||w^{(t)}||^2 \leq 2^d \cdot T \\ \Rightarrow T \leq \frac{2^d}{\gamma^2} = (4s+1)2^d = O(s \cdot 2^d) \end{aligned}$$

Which is the mistake bound for this kernel perceptron.

Problem 4

4.1

S = ((1,2),1,1), ((3,4),0,1), all the data are in the form $((x_1, x_2), A, Y)$.
For Demographic Parity: all Y here are 1, so the prediction is always 1.

$$\mathbb{P}_0[\hat{Y} = 1] = \mathbb{P}_1[\hat{Y} = 1] = 1$$

$$\mathbb{P}_0[\hat{Y} = 0] = \mathbb{P}_1[\hat{Y} = 0] = 0$$

For Equalized Odds: also prediction in any circumstance is 1 and $\mathbb{P}_a[Y = 0] = 0$,

$$\mathbb{P}_0[\hat{Y} = 1|Y = y] = \mathbb{P}_1[\hat{Y} = 1|Y = y] = 1$$

$$\mathbb{P}_0[\hat{Y} = 0|Y = y] = \mathbb{P}_1[\hat{Y} = 0|Y = y] = 0$$

For Predictive Parity, $\mathbb{P}_a[Y = 0] = 0$,

$$\mathbb{P}_0[Y = 1|\hat{Y} = \hat{y}] = \mathbb{P}_1[Y = 1|\hat{Y} = \hat{y}] = 1$$

$$\mathbb{P}_0[Y = 0|\hat{Y} = \hat{y}] = \mathbb{P}_1[Y = 0|\hat{Y} = \hat{y}] = 1$$

So the three fairness definitions are satisfied simultaneously.

4.2

If Demographic Parity holds, then

$$\mathbb{P}_0[\hat{Y} = \hat{y}] = \mathbb{P}_1[\hat{Y} = \hat{y}]$$

If Predictive Parity can hold at the same time, it means that:

$$\mathbb{P}_0[Y = y|\hat{Y} = \hat{y}] = \mathbb{P}_1[Y = y|\hat{Y} = \hat{y}]$$

$$\frac{\mathbb{P}_0[\hat{Y} = \hat{y}|Y = y] \times \mathbb{P}_0[Y = y]}{\mathbb{P}_0[\hat{Y} = \hat{y}]} = \frac{\mathbb{P}_1[\hat{Y} = \hat{y}|Y = y] \times \mathbb{P}_1[Y = y]}{\mathbb{P}_1[\hat{Y} = \hat{y}]}$$

Using Demographic Parity:

$$\mathbb{P}_0[\hat{Y} = \hat{y}|Y = y] \times \mathbb{P}_0[Y = y] = \mathbb{P}_1[\hat{Y} = \hat{y}|Y = y] \times \mathbb{P}_1[Y = y]$$

$$\mathbb{P}_0[\hat{Y} = \hat{y}, Y = y] = \mathbb{P}_1[\hat{Y} = \hat{y}, Y = y]$$

So,

$$\sum_{\hat{Y}} \mathbb{P}_0[\hat{Y} = \hat{y}, Y = y] = \sum_{\hat{Y}} \mathbb{P}_1[\hat{Y} = \hat{y}, Y = y]$$

$$\mathbb{P}_0[Y = y] = \mathbb{P}_1[Y = y]$$

But we know that A is dependent on Y, so

$$\mathbb{P}_0[Y = y] \neq \mathbb{P}_1[Y = y]$$

contradict with the conclusion above.
So if A is dependent on Y , then Demographic Parity and Predictive Parity cannot hold at the same time.

4.4

Claim:

$$FPR_a = \frac{p_a}{1-p_a} \cdot \frac{1-PPV_a}{PPV_a} \cdot (1 - FNR_a)$$

Proof:

$$\begin{aligned}
&\frac{\mathbb{P}_a[Y=1]}{1-\mathbb{P}_a[Y=1]} \frac{1-\mathbb{P}_a[Y=1|\hat{Y}=1]}{\mathbb{P}_a[Y=1|\hat{Y}=1]} (1-\mathbb{P}_a[\hat{Y}=0|Y=1]) \\
&= \frac{\mathbb{P}_a[Y=1]}{\mathbb{P}_a[Y=0]} \frac{\mathbb{P}_a[Y=0|\hat{Y}=1]}{\mathbb{P}_a[Y=1|\hat{Y}=1]} \mathbb{P}_a[\hat{Y}=1|Y=1] \\
&= \frac{\mathbb{P}_a[\hat{Y}=1, Y=1] \cdot \mathbb{P}_a[Y=0|\hat{Y}=1]}{\mathbb{P}_a[Y=0] \cdot \mathbb{P}_a[Y=1|\hat{Y}=1]} \\
&= \frac{\mathbb{P}_a[Y=1|\hat{Y}=1] \cdot \mathbb{P}_a[\hat{Y}=1] \cdot \mathbb{P}_a[Y=0|\hat{Y}=1]}{\mathbb{P}_a[Y=0] \cdot \mathbb{P}_a[Y=1|\hat{Y}=1]} \\
&= \frac{\mathbb{P}_a[Y=0|\hat{Y}=1] \cdot \mathbb{P}_a[\hat{Y}=1]}{\mathbb{P}_a[Y=0]} \\
&= \mathbb{P}_a[\hat{Y}=1|Y=0]
\end{aligned}$$

And our notions are

$$FPR_0 = \mathbb{P}_0[\hat{Y}=1|Y=0]$$
$$FPR_1 = \mathbb{P}_1[\hat{Y}=1|Y=0]$$

If Equalized Odds holds, it means that $FPR_0 = FPR_1$, similarly, $FNR_0 = FNR_1$.

$$\begin{aligned}
FPR_0 &= \frac{p_0}{1-p_0} \cdot \frac{1-PPV_0}{PPV_0} \cdot (1 - FNR_0) \\
&= FPR_1 = \frac{p_1}{1-p_1} \cdot \frac{1-PPV_1}{PPV_1} \cdot (1 - FNR_1)
\end{aligned}$$

As $FNR_0 = FNR_1$, $1 - FNR_0 = 1 - FNR_1$, so:

$$\frac{p_0}{1-p_0} \cdot \frac{1-PPV_0}{PPV_0} = \frac{p_1}{1-p_1} \cdot \frac{1-PPV_1}{PPV_1}$$

Because A is dependent on Y, so $p_0 \neq p_1$,

$$\begin{aligned}
\frac{1-PPV_0}{PPV_0} &\neq \frac{1-PPV_1}{PPV_1} \\
\frac{1}{PPV_0} - 1 &\neq \frac{1}{PPV_1} - 1 \\
\frac{1}{PPV_0} &\neq \frac{1}{PPV_1}
\end{aligned}$$

So, $PPV_0 \neq PPV_1$.
This proves that if A is dependent on Y, Equalized Odds and equality of Positive Predictive Value cannot hold at the same time.

Problem 5

5.1

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$\frac{\partial \sigma(x)}{\partial x} = \frac{1}{(1+e^{-x})^2} \cdot e^{-x}$$

$$1 - \sigma(x) = \frac{e^{-x}}{1+e^{-x}}$$

So,

$$\frac{\partial \sigma(x)}{\partial x} = \sigma(x)(1-\sigma(x))$$

5.2

In this problem we define $\odot$ as element-wise product (Hadamard product).

$$\begin{aligned}
\frac{\partial E}{\partial W_j} &= \frac{\partial E}{\partial \mathcal{F}} \cdot \frac{\partial \mathcal{F}}{\partial W_j} \\
&= \frac{1}{2n} \sum_{i=1}^{n} [2(\mathcal{F}(x_i) - y_i)]^T \left(\frac{\partial \sigma(W^T x_i + b)}{\partial (W^T x_i + b)} \cdot \frac{\partial (W^T x_i + b)}{\partial W_j} \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} (\mathcal{F}(x_i) - y_i)^T \mathbf{diag}(\sigma'((W^T x_i + b)_1), \ldots, \sigma'((W^T x_i + b)_{d_I})) \mathbf{diag}(x_{ij}, x_{ij}, \ldots, x_{ij}) \\
&= \frac{1}{n} \sum_{i=1}^{n} x_{ij} \left((\sigma(W^T x_i + b) - y_i) \odot \sigma'(W^T x_i + b) \right)^T
\end{aligned}$$

This is the ith line of $\frac{\partial E}{\partial W}$, thus we have

$$\begin{aligned}
\frac{\partial E}{\partial W} &= \frac{1}{n} \sum_{i=1}^{n} x_i \left((\sigma(W^T x_i + b) - y_i) \odot \sigma'(W^T x_i + b) \right)^T \\
&= \frac{1}{n} \sum_{i=1}^{n} x_i \left((\sigma(W^T x_i + b) - y_i) \odot \sigma(W^T x_i + b) \odot (1 - \sigma(W^T x_i + b)) \right)^T
\end{aligned}$$

$$\begin{aligned}
\frac{\partial E}{\partial b} &= \frac{\partial E}{\partial \mathcal{F}} \cdot \frac{\partial \mathcal{F}}{\partial b} \\
&= \frac{1}{2n} \sum_{i=1}^{n} [2(\mathcal{F}(x_i) - y_i)]^T \left(\frac{\partial \sigma(W^T x_i + b)}{\partial (W^T x_i + b)} \cdot \frac{\partial (W^T x_i + b)}{\partial b} \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} (\sigma(W^T x_i + b) - y_i)^T \left(\frac{\partial \sigma(W^T x_i + b)}{\partial (W^T x_i + b)} \right) \\
&= \frac{1}{n} \sum_{i=1}^{n} (\sigma(W^T x_i + b) - y_i) \odot \sigma(W^T x_i + b) \odot (1 - \sigma(W^T x_i + b))
\end{aligned}$$

5.3

a.

$$\frac{\partial \mathcal{N}(x)}{\partial x} = \frac{\partial (\sigma_1 \circ f_1 ... \sigma_n \circ f_n)}{\partial (f_1 \circ \cdots \circ \sigma_n \circ f_n)} \cdot \frac{\partial (f_1 \circ \sigma_2 \circ \cdots \circ \sigma_n \circ f_n)}{\partial (\sigma_2 \circ \cdots \circ \sigma_n \circ f_n)} \cdot \ldots \cdot \frac{\partial (\sigma_n \circ f_n)}{\partial f_n} \cdot \frac{\partial f_n}{\partial x}$$

We know that time complexity to compute any $f'(x)$ given x is $O(1)$, as well as to compute any $f(x)$ or $\sigma(x)$. If we compute this derivative sequentially, we need to calculate $f_1 \circ \cdots \circ \sigma_n \circ f_n$ first and get the first derivative term at this point, then throw all those results away. Then compute $\sigma_1 \circ \cdots \circ \sigma_n \circ f_n$, throw it away and move on to the next...
Therefore, when calculating the ith term, we need $O(n-i)$ time to calculate the point we are taking derivative at, and then $O(1)$ time to compute the derivative. So the total time complexity is $O(n-1) + O(n-2) + \cdots + O(1) = O(n^2)$.
And if we use another definition in the problem, i.e., $\mathcal{N}^L \circ \cdots \circ \mathcal{N}^1(x)$, we could have that computing $\mathcal{N}^i$ takes $O(i)$ time, and the time complexity is still $O(n^2)$. These two definitions seems different in indexing but the complexity remains the same here.

b. We can compute the derivations from the right most item, and save that value. For example, when we have calculated $f_n(x)$, it takes only $O(1)$ time to further compute $\sigma_n \circ f_n(x)$. Then if we do the computation from right to left, using the value that has already been computed to simplify the computation, the complexity of this algorithm will be reduced to $O(1) + O(1) + \cdots + O(1) = O(n)$.

5.4

Our code was submitted to Canvas. Here is our prediction.

 Pred  Label

Figure 1: Flower

 Pred  Label

Figure 2: Lincoin

5.5

The results in 5.4 are generated by the suggested network structure ($2 \times 128 \times 512 \times 1$ or 3), with Adam optimizer (learning rate=10^{-3}, $\beta_1 = 0.9, \beta_2 = 0.999$), batch size 256 after training over 250000 iterations. The training process takes about 18 minutes on CPU.
We've also tried other network structures. Here are some examples:

Network structure: $2 \times 64 \times 256 \times 1$ or 3

 Pred  Label

Figure 3: Flower

 Pred  Label

Figure 4: Lincoin

Network structure: $2 \times 32 \times 128 \times 1$ or 3

 Pred  Label

Figure 5: Flower

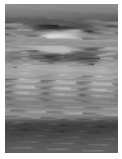 PredPred  Label

Figure 6: Lincoin

It can be inferred that smaller network width will result in weaker fitting performance when the other factors remain the same from these three set of results.

Also, we tried to remove one layer from the network. This network structure is $2 \times 128 \times 1$ or 3. We've modified the number of training iterations to ensure similar training time as previous experiments.

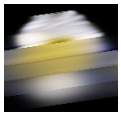 Pred  Label

Figure 7: Flower

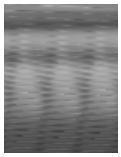 Pred  Label

Figure 8: Lincoin

As Lincoin's face is considered harder to learn than the flower's image even for the three-layer network, it makes sense that a small two-layer network cannot fit that function as we wish. And if we increase the hidden layer size to 512, then the prediction is

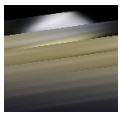 Pred  Label

Figure 9: Flower

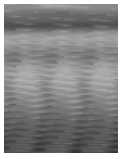 Pred  Label

Figure 10: Lincoin

This shows that removing one layer will severely damage the neural network's fitting ability. Even we increased the hidden layer's width and increased the training time, it still cannot capture the 'features' we want. What's worse, the 'fat' structure seems to make the network even harder to learn the target function for some reason.