Homework 1 - Machine Learning [redacted]
solution

Problem 1. (You can Google "monty hall problem" for more info)

First, we need to elaborate our assumptions: 1) gold is good, empty and tiger are bad; 2) the host will reveal the doors that are not the guest's first choice and not hiding gold in equal probability; 3) in the perspective of the guest, gold, empty and tiger distribute uniformly behind three doors.

Under those assumptions, we can have a simple solution:
The probability of the gold behind the firstly chosen door is $\frac{1}{3}$. And the guest doesn't switch and gets the gold.
The probability of the gold behind the other two doors that are not firstly chosen is $\frac{2}{3}$. And the host reveals that there is empty behind one of the two doors. So the probability of the guest switching and getting the gold is $\frac{2}{3}$.
So the optimal strategy is switching.

To show this process more formally and using Bayes' rules,
g as $gold$, e as $empty$, gu as $guest$, h as $host$, $d1$ as $door1$, $d2$ as $door2$ and $d3$ as $door3$.

For switching, the probability of getting gold is

$$
\begin{aligned}
&P(g = d2|gu = d1, h = d3, e = d3) \\
&= \frac{P(g = d2, gu = d1, h = d3, e = d3)}{P(gu = d1, h = d3, e = d3)} \\
&= \frac{P(h = d3|g = d2, gu = d1, e = d3)P(gu = d1|g = d2, e = d3)P(g = d2|e = d3)P(e = d3)}{P(h = d3|gu = d1, e = d3)P(gu = d1|e = d3)P(e = d3)}
\end{aligned}
$$

It is obvious that
$P(h = d3|g = d2, gu = d1, e = d3) = 1$,
$P(gu = d1|g = d2, e = d3) = P(gu = d1) = \frac{1}{3}$,
$P(g = d2|e = d3) = \frac{1}{2}$,
$P(e = d3) = \frac{1}{3}$,

$P(gu = d1|e = d3) = P(gu = d1) = \frac{1}{3}$.

$$\begin{aligned}
P(h = d3|gu = d1, e = d3) &= \sum_g P(h = d3, g|gu = d1, e = d3) \\
&= \sum_g P(h = d3|g, gu = d1, e = d3)P(g|gu = d1, e = d3) \\
&= \sum_g P(h = d3|g, gu = d1, e = d3)P(g|e = d3) \\
&= P(h = d3|g = d1, gu = d1, e = d3)P(g = d1|e = d3) \\
&\quad + P(h = d3|g = d2, gu = d1, e = d3)P(g = d2|e = d3) \\
&\quad + P(h = d3|g = d3, gu = d1, e = d3)P(g = d3|e = d3) \\
&= \frac{1}{2} \times \frac{1}{2} + 1 \times \frac{1}{2} + 0 \times 0 \\
&= \frac{3}{4}
\end{aligned}$$

So,

$$\begin{aligned}
P(g = d2|gu = d1, h = d3, e = d3) &= \frac{1 \times \frac{1}{3} \times \frac{1}{2} \times \frac{1}{3}}{\frac{3}{4} \times \frac{1}{3} \times \frac{1}{3}} \\
&= \frac{2}{3}
\end{aligned}$$

For not switching, the probability of getting gold is

$$\begin{aligned}
&P(g = d1|gu = d1, h = d3, e = d3) \\
&= \frac{P(g = d1, gu = d1, h = d3, e = d3)}{P(gu = d1, h = d3, e = d3)} \\
&= \frac{P(h = d3|g = d1, gu = d1, e.= d3)P(gu = d1|g = d1, e = d3)P(g = d1|e = d3)P(e = d3)}{P(h = d3|gu = d1, e = d3)P(gu = d1|e = d3)P(e = d3)}
\end{aligned}$$

It is obvious that
$P(h = d3|g = d1, gu = d1, e = d3) = \frac{1}{2}$,
$P(gu = d1|g = d1, e = d3) = P(gu = d1) = \frac{1}{3}$,
$P(g = d1|e = d3) = \frac{1}{2}$.

So,

$$\begin{aligned}
P(g = d1|gu = d1, h = d3, e = d3) &= \frac{\frac{1}{2} \times \frac{1}{3} \times \frac{1}{2} \times \frac{1}{3}}{\frac{3}{4} \times \frac{1}{3} \times \frac{1}{3}} \\
&= \frac{1}{3}
\end{aligned}$$

Thus, the optimal strategy is switching.

Problem 2:

2. Consider the following data set

Example	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

Find the smallest function that can accurately capture this data set. Start from just consider one feature to simple conjunctions and then expand to at least m-of-n rules. Explain why this problem is ill-posed.

Simple conjunctions:

Rule	Counter Example
$T \Leftrightarrow y$	#1
$x1 \Leftrightarrow y$	#3
$x2 \Leftrightarrow y$	#2
$x3 \Leftrightarrow y$	#1
$x4 \Leftrightarrow y$	#7
$x1 \wedge x2 \Leftrightarrow y$	#3
$x1 \wedge x3 \Leftrightarrow y$	#3
$x1 \wedge x4 \Leftrightarrow y$	#3
$x2 \wedge x3 \Leftrightarrow y$	#3
$x2 \wedge x4 \Leftrightarrow y$	#7
$x3 \wedge x4 \Leftrightarrow y$	#4
$x1 \wedge x2 \wedge x3 \Leftrightarrow y$	#3
$x1 \wedge x2 \wedge x4 \Leftrightarrow y$	#4
$x1 \wedge x3 \wedge x4 \Leftrightarrow y$	#3
$x2 \wedge x3 \wedge x4 \Leftrightarrow y$	#3
$x1 \wedge x2 \wedge x3 \wedge x4 \Leftrightarrow y$	#4

No simple rule could explain the data.

Expand to m of n rules using linear Threshold Units.

$$f(x) = \begin{cases} 1 & if\ w_1x_2 + w_1x_2 + \cdots + w_nx_n \geq w_0 \\ 0 & otherwise \end{cases}$$

X_1 X_3 X_4

i.e at least 2 of ~~{x1, x2, x3}~~ ⇔ y

1*x1+0*x2+1*x3+1*x4 ≥ 2

M of N rules	Counter Example			
Var	1 of	2 of	3 of	4 of
{x1}	#3			
{x2}	#2			
{x3}	#1			
{x4}	#7			
{x1, x2}	#3	#3		
{x1, x3}	#5	#3		
{x1, x4}	#6	#3		
{x2, x3}	#2	#3		
{x2, x4}	#2	#7		
{x3, x4}	#5	#4		
{x1, x2, x3}	#1	#3	#3	
{x1, x2, x4}	#2	#3	#3	
{x1, x3, x4}	#1	**Capture**	#3	
{x2, x3, x4}	#1	#5	#3	
{x1, x2, x3, x4}	#1	#5	#3	#3

One of the rules could capture the dataset.

The problem is ill-posed since 4 variables have 16 inputs. There will be total 2^{16} (65536) results of Boolean functions. We have 7 examples and have 2^9 possibilities of the rest datasets. The space of all possible function is too large making learning impossible. We have to apply our prior knowledge or experience to acquire a smaller hypothesis space. However, the solution is not unique, and the problem is ill-posed.

Problem 3. (You can also google "**One-VS-All**","**One-VS-One**" for further info)

For perceptron as a binary classifier, we have

$$h(\mathbf{x}) = sign(\mathbf{w}^{\mathrm{T}}\mathbf{x})$$

In canonical form, there is w_0 contained in $\mathbf{w}$ for the bias term.
To extend the binary classification to multi-class classification, we can simply calculate a weight parameter for each class. So our parameter for N classes becomes

$$\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_N]$$

The multi-class classifier becomes

$$h(\mathbf{x}) = [sign(\mathbf{w}_1^{\mathrm{T}}\mathbf{x}), sign(\mathbf{w}_2^{\mathrm{T}}\mathbf{x}), \cdots, sign(\mathbf{w}_N^{\mathrm{T}}\mathbf{x})]$$

So if the data point belongs to one of the classes, the sign of the corresponding class will be $+1$, the signs of all other classes will be -1.

Problem 4.

For the loss function

$$J(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} z_i \max(0, -y_i\mathbf{w}\cdot x_i)$$

We have

$$J_i(\mathbf{w}) = \max(0, -y_i\mathbf{w}\cdot x_i)$$

So

$$\frac{\partial J_i}{\partial \mathrm{w}_j} = \begin{cases} 0 & \text{if } y_i\mathbf{w}\cdot x_i > 0 \\ -z_i y_i x_{ij} & \text{otherwise} \end{cases}$$

$$\nabla J_i = \begin{cases} 0 & \text{if } y_i\mathbf{w}\cdot x_i > 0 \\ -z_i y_i x_i & \text{otherwise} \end{cases}$$

So the batch perceptron algorithm can be modified as following:
Let $\mathbf{w} = (0, 0, \cdots, 0)$ be the initial weight vector, $g = (0, 0, \cdots, 0)$ be the initial gradient vector.
Repeat until convergence
For $i = 1$ to N do

$u_i = \mathbf{w} \cdot x_i$

If $(y_i \cdot u_i < 0)$

For $j = 1$ to n do

If $(y_i = -1)$

$z_i = c_0$

If $(y_i = 1)$

$z_i = c_1$

$g_j := g_j - z_i y_i \cdot x_{ij}$

$g := g/N$

$\mathbf{w} := \mathbf{w} - \eta g$

Reset g to initial value

Problem 5. **(easy question)**

$$\begin{aligned} p_0(\mathbf{x}, \mathbf{w}) &= 1 - p_1(\mathbf{x}, \mathbf{w}) \\ &= 1 - \frac{1}{1 + exp(-\mathbf{w} \cdot \mathbf{x})} \\ &= \frac{exp(-\mathbf{w} \cdot \mathbf{x})}{1 + exp(-\mathbf{w} \cdot \mathbf{x})} \end{aligned}$$

So,

$$\begin{aligned} \log \frac{p_1(\mathbf{x}, \mathbf{w})}{p_0(\mathbf{x}, \mathbf{w})} &= \log \frac{\frac{1}{1+exp(-\mathbf{w}\cdot\mathbf{x})}}{\frac{exp(-\mathbf{w}\cdot\mathbf{x})}{1+exp(-\mathbf{w}\cdot\mathbf{x})}} \\ &= \log \frac{1}{exp(-\mathbf{w} \cdot \mathbf{x})} \\ &= -\log exp(-\mathbf{w} \cdot \mathbf{x}) \\ &= -(-\mathbf{w} \cdot \mathbf{x}) \\ &= \mathbf{w} \cdot \mathbf{x} \end{aligned}$$

Then,

$$
\begin{aligned}
p_1(\mathbf{x}, \mathbf{w}) &= \frac{1}{1 + exp(-\mathbf{w} \cdot \mathbf{x})} \\
&= \frac{1}{1 + e^{-\mathbf{w} \cdot \mathbf{x}}} \\
&= \frac{1}{1 + \left(e^{\mathbf{w} \cdot \mathbf{x}}\right)^{-1}} \\
&= \frac{1}{1 + \frac{1}{e^{\mathbf{w} \cdot \mathbf{x}}}} \\
&= \frac{1}{\frac{1 + e^{\mathbf{w} \cdot \mathbf{x}}}{e^{\mathbf{w} \cdot \mathbf{x}}}} \\
&= \frac{e^{\mathbf{w} \cdot \mathbf{x}}}{1 + e^{\mathbf{w} \cdot \mathbf{x}}} \\
&= \frac{exp(\mathbf{w} \cdot \mathbf{x})}{1 + exp(\mathbf{w} \cdot \mathbf{x})}
\end{aligned}
$$