CS221 Fall 2018 - 2019 Homework 1

Name: Dat Nguyen

By turning in this assignment, I agree by the Stanford honor code and declare that all of this is my own work.

Problem 1: Optimization and probability

(a)

$$
\begin{aligned}
f'(\theta) &= \sum_{i=1}^{n} w_i(\theta - x_i) \\
&= \theta \sum_{i=1}^{n} w_i - \sum_{i=1}^{n} w_i x_i \\
f'(\theta) &= 0 \\
\Leftrightarrow \theta &= \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i}
\end{aligned}
$$

(b) Let $s' = \text{argmax}_{s\in\{1,-1\}} \sum_{i=1}^{d} sx_i$. Then we have

$$\text{max}_{s\in\{1,-1\}} \sum_{i=1}^{d} sx_i = \sum_{i=1}^{d} s'x_i \leq= \sum_{i=1}^{d} \text{max}_{s\in\{1,-1\}} sx_i$$

(c) Let the expected value be V, the outcome at the i^{th} toss be s_i ($s_i \in \{1,2,3,4,5,6\}$) and the point we get at i^{th} toss be $R(s_i)$

$$R(s_i) = \begin{cases} 0 & \text{if } s_i \in \{1,3,4,5\} \\ -a & \text{if } s_i = 2 \\ b & \text{if } s_i = 6 \end{cases}$$

We have

$$
\begin{aligned}
V =& \sum_{s_1,s_2,\ldots} P(s_1,s_2,\ldots)(R(s_1)+R(s_2)+\ldots) \\
=& \sum_{s_1}\sum_{s_2,s_3,\ldots} P(s_1)P(s_2,s_3,\ldots|s_1)(R(s_1)+R(s_2)+\ldots) \\
=& \sum_{s_1}\sum_{s_2,s_3,\ldots} P(s_1)P(s_2,s_3,\ldots|s_1)(R(s_1)) + \sum_{s_1}\sum_{s_2,s_3,\ldots} P(s_1)P(s_2,s_3,\ldots|s_1)(R(s_2)+R(s_3)+\ldots) \\
=& \sum_{s_1} P(s_1)R(s_1) + \sum_{s_1\in\{2,3,4,5,6\}} P(s_1)P(s_2,s_3,\ldots|s_1)(R(s_2)+R(s_3)+\ldots)+ \\
& \sum_{s_1=1} P(s_1)P(s_2,s_3,\ldots|s_1)(R(s_2)+R(s_3)+\ldots) \\
=& \frac{a+b}{6} + \frac{5}{6}V + 0
\end{aligned}
$$

Therefore

$$
\begin{aligned}
\frac{V}{6} &= \frac{a+b}{6} \\
V &= a+b
\end{aligned}
$$

(d)

$$
\begin{aligned}
\mathrm{log}L(p) &= \mathrm{log}(p^4(1-p)^3) \\
&= 4\mathrm{log}p + 3\mathrm{log}(1-p)
\end{aligned}
$$

So

$$
\begin{aligned}
\frac{\partial \mathrm{log}L(p)}{\partial p} &= \frac{4}{p} - \frac{3}{1-p} \\
&= \frac{4-7p}{p(1-p}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathrm{log}L(p)}{\partial p} &= 0 \\
\Leftrightarrow p &= \frac{4}{7}
\end{aligned}
$$

(e)

$$
\begin{aligned}
\Delta f(w) &= \sum_{i=1}^{n}\sum_{j=1}^{n} 2(\mathbf{a}_i^T\mathbf{w} - \mathbf{b}_j^T\mathbf{w})\Delta(\mathbf{a}_i^T\mathbf{w} - \mathbf{b}_j^T\mathbf{w}) - 2\lambda w \\
&= \mathbf{w}\sum_{i=1}^{n}\sum_{j=1}^{n} 2(\mathbf{a}_i^T - \mathbf{b}_j^T)(\mathbf{a}_i - \mathbf{b}_j) - 2\lambda w
\end{aligned}
$$

Problem 2: Complexity

(a) For each part of the face, we first choose the top left corner then choose the width and height. since there are n^2 way to choose top left corner, up to n way to choose width and height, in total we have up to $n^2 nn = n^4$ ways to place a part of the face. because the face has 6 parts, asymptotically we have $(n^4)^6 = n^{24}$ ways to represent the face.

(b) We will adopt a dynamic programming approach, let $v[i, j]$ be the optimal cost to reach position i, j.

```
let v[0, j] = 0 for all j and v[i, 0] = 0 for all i
for i = 1 to n
  for j = 1 to n
     v[i, j] = c(i, j) + max(v[i - 1, j], v[i, j - 1])
return v[n, n]
```

the running time of the above algorithm is $o(n^2)$

(c) From step 1 to step n - 1, at each step we have 2 choices of actions i) stop at this step to prepare for the next move or ii) continue this step as a step of the current move. when we reach step n we are done so we do not need to choose actions. therefore there are 2^{n-1} ways to reach the top.

(d) We have

$$
\begin{aligned}
f(\mathbf{w}) &= \sum_{i=1}^{n}\sum_{j=1}^{n}(\mathbf{a_i}^T\mathbf{w} - \mathbf{b_j}^T\mathbf{w})^2 + \lambda||\mathbf{w}||_2^2 \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}(\mathbf{w}^T\mathbf{a_i} - \mathbf{w}^T\mathbf{b_j})(\mathbf{a_i}^T\mathbf{w} - \mathbf{b_j}^T\mathbf{w}) + \lambda||\mathbf{w}||_2^2 \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}(\mathbf{w}^T\mathbf{a_i}\mathbf{a_i}^T\mathbf{w} - \mathbf{w}^T\mathbf{a_i}\mathbf{b_j}^T\mathbf{w} - \mathbf{w}^T\mathbf{b_j}\mathbf{a_i}^T\mathbf{w} + \mathbf{w}^T\mathbf{b_j}\mathbf{b_j}^T\mathbf{w}) + \lambda||\mathbf{w}||_2^2 \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{w}^T(\mathbf{a_i}\mathbf{a_i}^T - 2\mathbf{a_i}\mathbf{b_j}^T + \mathbf{b_j}\mathbf{b_j}^T)\mathbf{w} + \lambda||\mathbf{w}||_2^2 \\
&= \mathbf{w}^T\Big(\sum_{i=1}^{n}\sum_{j=1}^{n}(\mathbf{a_i}\mathbf{a_i}^T - 2\mathbf{a_i}\mathbf{b_j}^T + \mathbf{b_j}\mathbf{b_j}^T)\Big)\mathbf{w} + \lambda||\mathbf{w}||_2^2 \\
&= \mathbf{w}^T\Big(\sum_{i=1}^{n}n\mathbf{a_i}\mathbf{a_i}^T - 2\sum_{i=1}^{n}\mathbf{a_i}\sum_{j=1}^{n}\mathbf{b_j}^T + \sum_{j=1}^{n}n\mathbf{b_j}\mathbf{b_j}^T\Big)\mathbf{w} + \lambda||\mathbf{w}||_2^2
\end{aligned}
$$

In the processing step we will calculate

$$A = \big(\sum_{i=1}^{n} n\mathbf{a_i}\mathbf{a_i}^T - 2\sum_{i=1}^{n}\mathbf{a_i}\sum_{j=1}^{n}\mathbf{b_j}^T + \sum_{j=1}^{n} n\mathbf{b_j}\mathbf{b_j}^T\big)$$

Because $\sum_{i=1}^{n} n\mathbf{a_i}\mathbf{a_i}^T$ takes $O(nd^2)$, $2\sum_{i=1}^{n}\mathbf{a_i}\sum_{j=1}^{n}\mathbf{b_j}^T$ takes $O(nd)$ and $\sum_{j=1}^{n} n\mathbf{b_j}\mathbf{b_j}^T$ takes $O(nd^2)$, the processing step takes time $O(nd^2)$.

For any given $\mathbf{w}$ we compute $\mathbf{w}^T A\mathbf{w}$ which takes time $O(d^2)$ and $\lambda||\mathbf{w}||_2^2$ which also takes time $O(d^2)$.