Homework 1

Benson Huang

The Learning Problem

1.

[a] and [b] have no particular pattern to learn and [c] is programmable. Hence we choose [d].

2.

The methods in [a]$\sim$[d] don't contain experience accumulated/computed from data. So we choose [e].

Perceptron Learning Algorithm

3.

Define $R^2 = \max\limits_n \|\boldsymbol{x}_n\|^2$ and $\rho = \min\limits_n y_n \frac{\boldsymbol{w}_f^T}{\|\boldsymbol{w}_f\|}\boldsymbol{x}_n$. From page 16 of lecture 2, we know that the PLA will halt before $T = R^2/\rho^2$, Therefore, if we scale down all x_n, the upper bound of execution time will become

$$\begin{aligned} T' &= \max_n \|\frac{1}{4}\boldsymbol{x}_n\|^2 / (\min_n y_n \frac{\boldsymbol{w}_f^T}{\|\boldsymbol{w}_f\|}\frac{1}{4}\boldsymbol{x}_n)^2 = \frac{1}{16}\max_n \|\boldsymbol{x}_n\|^2/(\frac{1}{4}\min_n y_n \frac{\boldsymbol{w}_f^T}{\|\boldsymbol{w}_f\|}\boldsymbol{x}_n)^2 \\ &= R^2/\rho^2 = T. \end{aligned}$$

Hence, the worst-case speed of PLA is unchanged.

4.

Since $\eta_t = \frac{1}{\|\boldsymbol{x}_{n(t)}\|}$, we have

$$\begin{aligned} \|\boldsymbol{w}_t\|^2 &= \left\|\boldsymbol{w}_{t-1} + y_{n(t)}\frac{\boldsymbol{x}_{n(t)}}{\|\boldsymbol{x}_{n(t)}\|}\right\|^2 \le \|\boldsymbol{w}_{t-1}\|^2 + \max_n \left\|y_n \frac{\boldsymbol{x}_n}{\|\boldsymbol{x}_n\|}\right\|^2 \\ &\le \|\boldsymbol{w}_{t-1}\|^2 + 1 \le \cdots \le \|\boldsymbol{w}_0\|^2 + t \\ &= t \end{aligned} \tag{1}$$

and

$$\begin{aligned} \boldsymbol{w}_f^T\boldsymbol{w}_t &= \boldsymbol{w}_f^T \cdot (\boldsymbol{w}_{t-1} + \eta_{t-1}y_{n(t-1)}\boldsymbol{x}_{n(t-1)}) \\ &= \boldsymbol{w}_f^T\boldsymbol{w}_{t-1} + y_{n(t-1)}\boldsymbol{w}_f^T\frac{\boldsymbol{x}_{n(t-1)}}{\|\boldsymbol{x}_{n(t)}\|} \\ &\ge \boldsymbol{w}_f^T\boldsymbol{w}_{t-1} + \min_n y_n \boldsymbol{w}_f^T \frac{\boldsymbol{x}_n}{\|\boldsymbol{x}_n\|} \\ &\ge \cdots \ge \boldsymbol{w}_f^T\boldsymbol{w}_0 + t\min_n y_n \boldsymbol{w}_f^T \frac{\boldsymbol{x}_n}{\|\boldsymbol{x}_n\|} \\ &= t\min_n y_n \boldsymbol{w}_f^T \frac{\boldsymbol{x}_n}{\|\boldsymbol{x}_n\|}. \end{aligned} \tag{2}$$

Therefore,

$$\begin{aligned} \frac{|\boldsymbol{w}_f^T\boldsymbol{w}_t|}{\|\boldsymbol{w}_f\|\|\boldsymbol{w}_t\|} &\overset{(2)}{\ge} \frac{t\cdot\min_n y_n \boldsymbol{w}_f^T \frac{\boldsymbol{x}_n}{\|\boldsymbol{x}_n\|}}{\|\boldsymbol{w}_f\|\|\boldsymbol{w}_t\|} \\ &\overset{(1)}{\ge} \frac{t\cdot\min_n y_n \boldsymbol{w}_f^T \frac{\boldsymbol{x}_n}{\|\boldsymbol{x}_n\|}}{\sqrt{t}\cdot\|\boldsymbol{w}_f\|} \\ &= \sqrt{t}\min_n \frac{|\boldsymbol{w}_f^T\boldsymbol{x}_n|}{\|\boldsymbol{w}_f\|\|\|\boldsymbol{x}_n\|}. \end{aligned} \tag{3}$$

Since $\frac{|\boldsymbol{w}_f^T \boldsymbol{w}_t|}{\|\boldsymbol{w}_f\|\|\boldsymbol{w}_t\|}$ must be not greater than 1, PLA must halt when $t = (\min_n \frac{|\boldsymbol{w}_f^T \boldsymbol{x}_n|}{\|\boldsymbol{w}_f\|\|\boldsymbol{x}_n\|})^{-2} = \rho^{-2}$.

5.

Suppose $y_{n(t)}\boldsymbol{w}_t^T\boldsymbol{x}_{n(t)} \leq 0$ and we hope to correct the mistake by $\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + \eta_t y_{n(t)}\boldsymbol{x}_{n(t)}$. Now we require $y_{n(t)}\boldsymbol{w}_{t+1}^T\boldsymbol{x}_{n(t)} > 0$, that is,

$$
\begin{aligned}
& y_{n(t)}\boldsymbol{w}_{t+1}\boldsymbol{x}_{n(t)} = y_{n(t)}(\boldsymbol{w}_t + \eta_t y_{n(t)}\boldsymbol{x}_{n(t)})^T\boldsymbol{x}_{n(t)} > 0 \\
\iff & y_{n(t)}(\eta_t y_{n(t)}\boldsymbol{x}_{n(t)})^T\boldsymbol{x}_{n(t)} + y_{n(t)}\boldsymbol{w}_t^T\boldsymbol{x}_{n(t)} > 0 \\
\iff & \eta_t\|x_{n(t)}\|^2 + y_{n(t)}\boldsymbol{w}_t^T\boldsymbol{x}_{n(t)} > 0 \\
\iff & \eta_t > -\frac{y_{n(t)}\boldsymbol{w}_t^T\boldsymbol{x}_{n(t)}}{\|\boldsymbol{x}_{n(t)}\|^2}.
\end{aligned} \tag{4}
$$

Then

$$
\begin{aligned}
y_{n(t)}\boldsymbol{w}_{t+1}^T\boldsymbol{x}_{n(t)} &= y_{n(t)}\left(\boldsymbol{w}_t + \eta_t y_{n(t)}\boldsymbol{x}_{n(t)}\right)^T\boldsymbol{x}_{n(t)} \\
&= y_{n(t)}\left(\boldsymbol{w}_t + \left\lfloor -\frac{y_{n(t)}\boldsymbol{w}_t^T\boldsymbol{x}_{n(t)}}{\|\boldsymbol{x}_{n(t)}\|^2} + 1\right\rfloor y_{n(t)}\boldsymbol{x}_{n(t)}\right)^T\boldsymbol{x}_{n(t)} \\
&\overset{(4)}{>} 0.
\end{aligned} \tag{5}
$$

6.

For [a] and [b] in **5.**, we can show that if we update the weights by $\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + \eta y_{n(t)}\boldsymbol{x}_{n(t)},\ \eta > 0$ and the data is separable, PLA will halt with a perfect line eventually:

Proof. Since $y_{n(t)}\boldsymbol{w}_t^T\boldsymbol{x}_{n(t)} \leq 0$, we have

$$
\begin{aligned}
\|\boldsymbol{w}_t\|^2 &= \left\|\boldsymbol{w}_{t-1} + \eta y_{n(t)}\boldsymbol{x}_{n(t)}\right\|^2 \\
&\leq \|\boldsymbol{w}_{t-1}\|^2 + 2\eta\boldsymbol{w}_{t-1}^T\boldsymbol{x}_{n(t)} + \eta^2\|\boldsymbol{x}_{n(t)}\|^2 \\
&\leq \|\boldsymbol{w}_{t-1}\|^2 + \eta^2\|\boldsymbol{x}_{n(t)}\|^2 \leq \|\boldsymbol{w}_{t-1}\|^2 + \eta^2\min_n\|\boldsymbol{x}_n\|^2 \\
&\leq \cdots \leq t\eta^2\min_n\|\boldsymbol{x}_n\|^2
\end{aligned} \tag{6}
$$

and

$$
\begin{aligned}
\boldsymbol{w}_f^T\boldsymbol{w}_t &= \boldsymbol{w}_f^T \cdot (\boldsymbol{w}_{t-1} + \eta y_{n(t-1)}\boldsymbol{x}_{n(t-1)}) \\
&= \boldsymbol{w}_f^T\boldsymbol{w}_{t-1} + \eta y_{n(t-1)}\boldsymbol{w}_f^T\boldsymbol{x}_{n(t-1)} \\
&\geq \boldsymbol{w}_f^T\boldsymbol{w}_{t-1} + \eta\min_n y_n\boldsymbol{w}_f^T\boldsymbol{x}_n \\
&\geq \cdots \geq \boldsymbol{w}_f^T\boldsymbol{w}_0 + t\eta\min_n y_n\boldsymbol{w}_f^T\boldsymbol{x}_n \\
&= t\eta\min_n y_n\boldsymbol{w}_f^T\boldsymbol{x}_n.
\end{aligned} \tag{7}
$$

Hence,

$$
\begin{aligned}
\frac{|\boldsymbol{w}_f^T\boldsymbol{w}_t|}{\|\boldsymbol{w}_f\|\|\boldsymbol{w}_t\|} &\overset{(7)}{\geq} \frac{t\eta\cdot\min_n y_n\boldsymbol{w}_f^T\boldsymbol{x}_n}{\|\boldsymbol{w}_f\|\|\boldsymbol{w}_t\|} \\
&\overset{(6)}{\geq} \frac{t\eta\cdot\min_n y_n\boldsymbol{w}_f^T\boldsymbol{x}_n}{\sqrt{t}\eta\sqrt{\min_n\|\boldsymbol{x}_n\|^2}} \\
&= \sqrt{t}\frac{\min_n y_n\boldsymbol{w}_f^T\boldsymbol{x}_n}{\sqrt{\min_n\|\boldsymbol{x}_n\|^2}}.
\end{aligned}
$$

Since $\frac{|\boldsymbol{w}_f^T\boldsymbol{w}_t|}{\|\boldsymbol{w}_f\|\|\boldsymbol{w}_t\|}$ must be not greater than 1, PLA must halt when reach $\frac{\min_n\|\boldsymbol{x}_n\|^2}{(\min_n y_n\boldsymbol{w}_f^T\boldsymbol{x}_n)^2}$, which means we eventually find a perfect line (No instance $\boldsymbol{x}_n$ causes $y_{n(t)}\boldsymbol{w}_t^T\boldsymbol{x}_n \leq 0$). ☐

For [c] in **5.**, consider that we have only one single training instance $(\boldsymbol{x}_1, y_1) = ((1,0,0)^T, 1)$ in the dataset (therefore, it's trivially separable). Now we start from $\boldsymbol{w}_0 = (0,0,0)^T$. Clearly, $y_1 \boldsymbol{w}_t^T \boldsymbol{x}_1 \leq 0$. Since the PLA updates the weights by

$$\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + (-\frac{y_{n(t)}\boldsymbol{w}_t^T \boldsymbol{x}_{n(t)}}{\|\boldsymbol{x}_{n(t)}\|^2}) y_{n(t)} \boldsymbol{x}_{n(t)},$$

we have

$$\boldsymbol{w}_1 \leftarrow (0,0,0)^T + (-\frac{(0,0,0)^T \cdot (1,0,0)}{1})(1,0,0)^T = (0,0,0)^T.$$

Obviously, $\boldsymbol{w}_t$ will always be $(0,0,0)^T$ and thus the PLA will never halt. For [d] in **5.**, we need to bound the growth rate of $\boldsymbol{w}_f \boldsymbol{w}_t$ and $\|\boldsymbol{w}_t\|$:
Since $y_n \boldsymbol{w}_f \boldsymbol{x}_n > 0,\ \forall \boldsymbol{x}_n$ in the dataset and $y_n \boldsymbol{w}_t^T \boldsymbol{x}_{n(t)} \leq 0$, we have

$$\begin{aligned}
\boldsymbol{w}_f^T \boldsymbol{w}_t &= \boldsymbol{w}_f^T \left(\boldsymbol{w}_{t-1} + \left\lfloor -\frac{y_{n(t-1)}\boldsymbol{w}_{t-1}^T \boldsymbol{x}_{n(t-1)}}{\|\boldsymbol{x}_{n(t-1)}\|^2} + 1 \right\rfloor y_{n(t-1)} \boldsymbol{x}_{n(t-1)} \right) \\
&= \boldsymbol{w}_f^T \boldsymbol{w}_{t-1} + \left\lfloor 1 - \frac{y_{n(t-1)}\boldsymbol{w}_{t-1}^T \boldsymbol{x}_{n(t-1)}}{\|\boldsymbol{x}_{n(t-1)}\|^2} \right\rfloor y_{n(t-1)} \boldsymbol{w}_f^T \boldsymbol{x}_{n(t-1)} \\
&\geq \boldsymbol{w}_f^T \boldsymbol{w}_{t-1} + y_{n(t-1)} \boldsymbol{w}_f^T \boldsymbol{x}_{n(t-1)} \\
&\geq \boldsymbol{w}_f^T \boldsymbol{w}_{t-1} + \min_n y_n \boldsymbol{w}_f^T \boldsymbol{x}_n \\
&\geq \cdots \geq t \min_n y_n \boldsymbol{w}_f^T \boldsymbol{x}_n.
\end{aligned}$$

If we let ρ denote $\min_n y_n \frac{\boldsymbol{w}_f^T}{\|\boldsymbol{w}_f\|} \boldsymbol{x}_n$, then we get

$$\frac{\boldsymbol{w}_f^T \boldsymbol{w}_t}{\|\boldsymbol{w}_f\|} \geq t\rho. \tag{8}$$

On the other hand, to make the analysis easier, we suppose that

$$\left\lfloor -\frac{y_{n(t-1)}\boldsymbol{w}_{t-1}^T \boldsymbol{x}_{n(t-1)}}{\|\boldsymbol{x}_{n(t-1)}\|^2} + 1 \right\rfloor = \left(-\frac{y_{n(t-1)}\boldsymbol{w}_{t-1}^T \boldsymbol{x}_{n(t-1)}}{\|\boldsymbol{x}_{n(t-1)}\|^2} + 1 \right).$$

Notice that his assumption will lead to a looser upper bound of $\|w_t\|$, which can be easily proved by induction. Now we can decompose $\boldsymbol{w}_t$ into to two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$:

$$\begin{aligned}
\boldsymbol{w}_t &= \boldsymbol{w}_{t-1} + \left(-\frac{y_{n(t-1)}\boldsymbol{w}_{t-1}^T \boldsymbol{x}_{n(t-1)}}{\|\boldsymbol{x}_{n(t-1)}\|^2} + 1 \right) y_{n(t-1)} \boldsymbol{x}_{n(t-1)} \\
&= \underbrace{\left(\boldsymbol{w}_{t-1} - \frac{\langle \boldsymbol{w}_{t-1}, \boldsymbol{x}_{n(t-1)} \rangle}{\|\boldsymbol{x}_{n(t-1)}\|^2} \boldsymbol{x}_{n(t-1)} \right)}_{\boldsymbol{u}} + \underbrace{y_{n(t-1)} \boldsymbol{x}_{n(t-1)}}_{\boldsymbol{v}}.
\end{aligned}$$

Also notice that

$$\boldsymbol{w}_{t-1} = \boldsymbol{u} + \frac{\langle \boldsymbol{w}_{t-1}, \boldsymbol{x}_{n(t-1)} \rangle}{\|\boldsymbol{x}_{n(t-1)}\|^2} \boldsymbol{x}_{n(t-1)}$$

and

$$\begin{aligned}
&\langle \boldsymbol{u}, \boldsymbol{x}_{n(t-1)} \rangle = \langle \boldsymbol{w}_{t-1}, \boldsymbol{x}_{n(t-1)} \rangle - \langle \boldsymbol{w}_{t-1}, x_{n(t-1)} \rangle \frac{\langle \boldsymbol{x}_{n(t-1)}, \boldsymbol{x}_{n(t-1)} \rangle}{\|\boldsymbol{x}_{n(t-1)}\|^2} = 0 \\
\Rightarrow\quad & \boldsymbol{u} \perp \frac{\langle \boldsymbol{w}_{t-1}, \boldsymbol{x}_{n(t-1)} \rangle}{\|\boldsymbol{x}_{n(t-1)}\|^2} \boldsymbol{x}_{n(t-1)}.
\end{aligned}$$

Thus, we can apply Pythagorean theorem to obtain

$$\|w_{t-1}\|^2 = \|\boldsymbol{u}\|^2 + \left\| \frac{\langle \boldsymbol{w}_{t-1}, \boldsymbol{x}_{n(t-1)} \rangle}{\|\boldsymbol{x}_{n(t-1)}\|^2} \boldsymbol{x}_{n(t-1)} \right\|^2 \geq \|\boldsymbol{u}\|^2. \tag{9}$$

Also notice that

$$\begin{aligned}\langle \boldsymbol{u}, \boldsymbol{v}\rangle &= \left\langle \boldsymbol{w}_{t-1} - \frac{\langle \boldsymbol{w}_{t-1}, \boldsymbol{x}_{n(t-1)}\rangle}{\|\boldsymbol{x}_{n(t-1)}\|^2}\boldsymbol{x}_{n(t-1)}, y_{n(t-1)}\boldsymbol{x}_{n(t-1)} \right\rangle \\ &= \left\langle \boldsymbol{w}_{t-1}, y_{n(t-1)}\boldsymbol{x}_{n(t-1)}\right\rangle - \left\langle \boldsymbol{w}_{t-1}, y_{n(t-1)}\boldsymbol{x}_{n(t-1)}\right\rangle \frac{\langle \boldsymbol{x}_{n(t-1)}, \boldsymbol{x}_{n(t-1)}\rangle}{\|\boldsymbol{x}_{n(t-1)}\|^2} \\ &= 0.\end{aligned}$$

Hence, $\boldsymbol{u} \perp \boldsymbol{v}$ and we can apply Pythagorean theorem again (Let R^2 denote $\max_n \|\boldsymbol{x}_n\|^2$):

$$\begin{aligned}\|\boldsymbol{w}_t\|^2 &= \|\boldsymbol{u}\|^2 + \|\boldsymbol{v}\|^2 \\ &\overset{(9)}{\leq} \|\boldsymbol{w}_{t-1}\|^2 + \|\boldsymbol{v}\|^2 \\ &\leq \|\boldsymbol{w}_{t-1}\|^2 + R^2 \\ &\leq \cdots \leq tR^2. \end{aligned} \tag{10}$$

The upper bound in (10) is trivially the upper bound of $\|w_t\|$ if $\left\lfloor -\frac{y_{n(t-1)}\boldsymbol{w}_{t-1}^T\boldsymbol{x}_{n(t-1)}}{\|\boldsymbol{x}_{n(t-1)}\|^2} + 1 \right\rfloor < -\frac{y_{n(t-1)}\boldsymbol{w}_{t-1}^T\boldsymbol{x}_{n(t-1)}}{\|\boldsymbol{x}_{n(t-1)}\|^2} + 1$. Combine with (9) and (10), we can repeat the same process on page 16 of lecture 2 to show that this PLA will halt before t reach R^2/ρ^2.

For [d] in **6.**, consider that we have only one single training instance $(\boldsymbol{x}_1, y_1) = ((1,0,0)^T, 1)$ in the dataset (therefore, it's trivially separable). Now we start from $\boldsymbol{w}_0 = (0,0,0)^T$. Clearly, $y_1\boldsymbol{w}_t^T\boldsymbol{x}_1 \leq 0$. So the PLA updates the weights by

$$\boldsymbol{w}_1 \leftarrow \boldsymbol{w}_0 - \left\lfloor -\frac{y_1\boldsymbol{w}_0^T\boldsymbol{x}_1}{\|\boldsymbol{x}_1\|^2} + 1 \right\rfloor y_1\boldsymbol{x}_1.$$

Therefore, $\boldsymbol{w}_1$ becomes $(-1,0,0)^T$. It's obvious that $y_1\boldsymbol{w}_1^T\boldsymbol{x}_1 < 0$. Therefore, the algorithm will continue to compute $\boldsymbol{w}_2 = (-1,0,0)^T - 2(1,0,0)^T = (-3,0,0)^T$, and so on. In this way, we can easily proved that $\boldsymbol{w}_t = (2^t - 1)\boldsymbol{w}_1$ by induction. Thus, the algorithm will never halt. From the discussion among the choices in **5.**, we conclude that only [a], [b] and [d] are guaranteed to halt.

From the discussion among the choices in **5.**, we conclude that only [a], [b] and [d] are guaranteed to halt.

Types of Learning

7.

We choose [e] because the feature reinforcement learning is based its interaction with the environment. This is a demonstration of self-practicing.

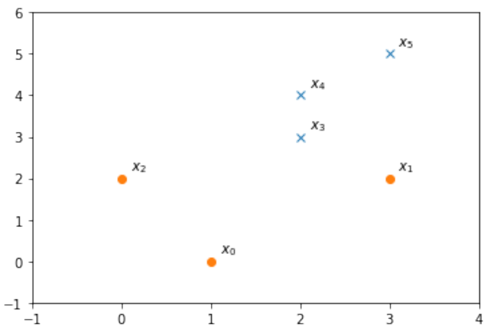
8.

The input data of this model is not preprocessed, so it's raw data. Since some training examples are with label while some are not, so it's a semi-supervised learning algorithm.

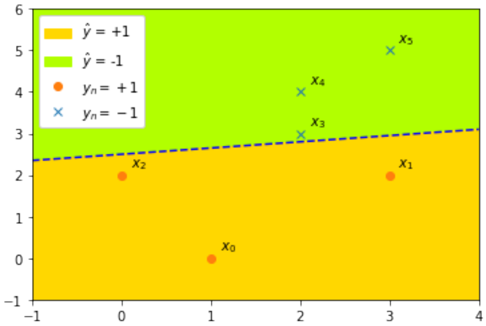
Off-Training-Set Error

9.

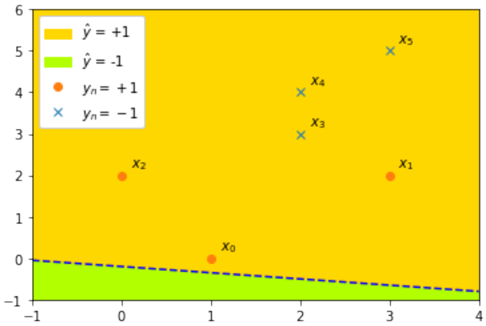
First we plot all the training example:



Set $(w_0, w_1, w_2) = (2.5, 0.15, -1)$, we can create a perceptron with $E_{\text{in}}(h) = E_{\text{out}}(h) = 0$:



Set $(w_0, w_1, w_2) = (0.2, 0.15, 1)$, we can create a perceptron with $E_{\text{in}}(h) = 0$ but $E_{\text{out}}(h) = 1$:



(See Appendix for the codes.)

Hoeffding Inequality

10.

Suppose we have tossed the biased coin N times and we want to guess which side of the coin has a greater chance of coming up. An intuitive way is to choose the face with higher frequency, denoted by ν (> 0.5), of coming up during the N trials.

Now, we want to evaluate how likely we choose the wrong face. Without loss of generality, we assume $\epsilon > 0$. If we have guessed wrong (i.e. the true probability μ of this face coming up is $0.5 - \epsilon < 0.5$), then we have

$$\nu > 0.5$$

but

$$\mu = 0.5 - \epsilon.$$

That is,

$$\nu - \mu > \epsilon.$$

Therefore, applying Hoeffding's inequality, we have

$$\begin{align} 1 - \mathbb{P}[\text{ choosing right face }] &= \mathbb{P}[\text{ choosing wrong face }] \tag{11} \\ &\leq \mathbb{P}[|\nu - \mu| > \epsilon] \tag{12} \\ &\leq 2\exp(-2\epsilon^2 N) \tag{13} \end{align}$$

Equivalently,

$$\mathbb{P}[\text{ choosing right face }] \geq 1 - 2\exp(-2\epsilon^2 N). \tag{14}$$

Hence, to ensure that $\mathbb{P}[\text{ choosing right face }] \geq 1 - \delta$, the following condition must be meet:

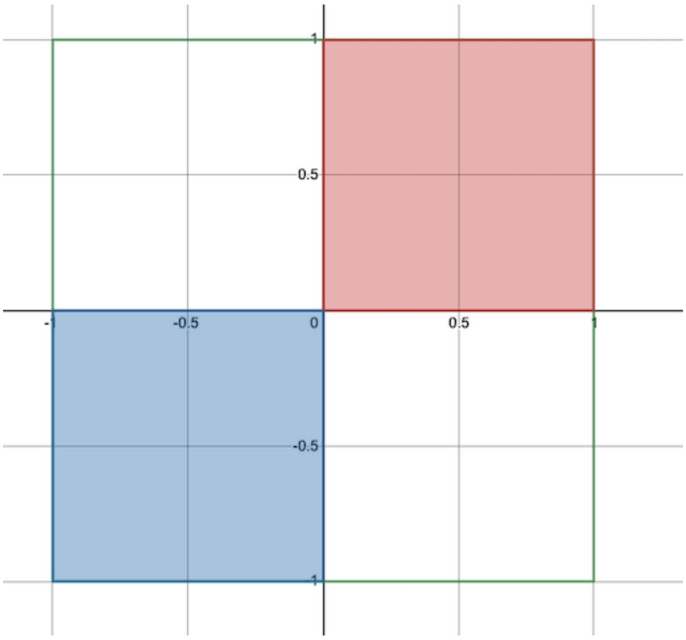
$$\begin{align} 2\exp(-2\epsilon^2 N) &\leq \delta \tag{15} \\ \iff \log 2 - 2\epsilon^2 N &\leq \log \delta \tag{16} \\ \iff N &\geq \frac{1}{2\epsilon^2} \log \frac{2}{\delta}. \tag{17} \end{align}$$

So we need to tossed the coin at least $\frac{1}{2\epsilon^2} \log \frac{2}{\delta}$ times.

Bad Data

11.

First we plot the region where $h_2(x) = f(x)$:



The red-shaded region with area 1 is where $h_2(x) = f(x) = +1$ and the blue-shaded region with area 1 is where $h_2(x) = f(x) = -1$. Since the total area that may be sampled is 4, we have

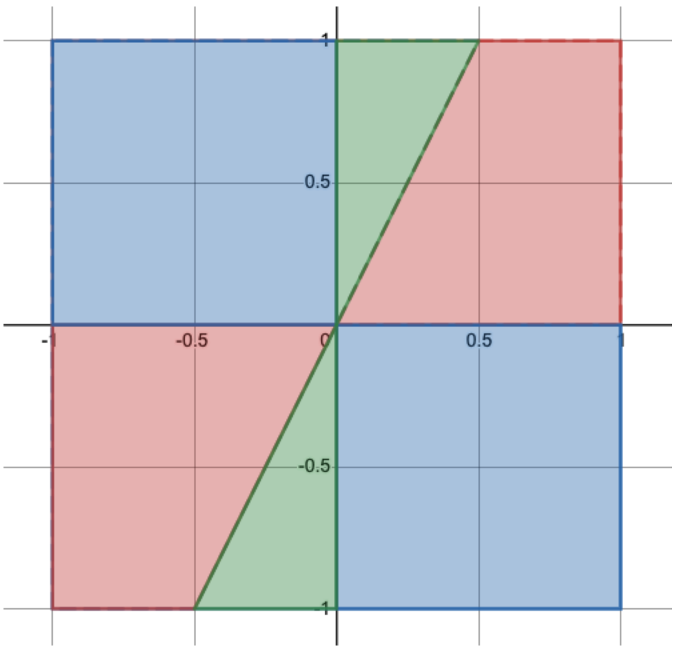
$$\mathbb{P}[h_2(\boldsymbol{x}_i) = f(\boldsymbol{x}_i)] = \mathbb{P}[h_2(\boldsymbol{x}) = f(\boldsymbol{x}) = +1] + \mathbb{P}[h_2(\boldsymbol{x}) = f(\boldsymbol{x}) = -1] = \frac{1}{2},\ i = 1, ..., 5.$$

Since the training example are drawing independently from $[-1, 1] \times [-1, 1]$,

$$\mathbb{P}[E_{\text{in}}(h_2)] = \prod_{i=1}^{5} \mathbb{P}[h_2(\boldsymbol{x}_i) = f(\boldsymbol{x}_i)] = \frac{1}{2^5}.$$

12.

First we plot the region where $h_1(\boldsymbol{x}) = h_2(\boldsymbol{x})$:



From the figure we can compute the probability of the following 3 events:

1. the probability of h_1 and h_2 making the same prediction (red-shaded area):

$$\mathbb{P}[h_1(\boldsymbol{x}_i) = h_2(\boldsymbol{x}_i)] = \mathbb{P}[h_1(\boldsymbol{x}_i) = h_2(\boldsymbol{x}_i) = +1] + P[h_1(\boldsymbol{x}_i) = h_2(\boldsymbol{x}_i) = -1] = (\frac{3}{4} + \frac{3}{4})/2 = \frac{3}{8}$$

2. the probability of h_1 making wrong prediction but h_2 making right prediction (green prediction):

$$\mathbb{P}[h_1(\boldsymbol{x}_i) \neq h_2(\boldsymbol{x}_i) = f(\boldsymbol{x}_i)] = \frac{1}{8}$$

3. the probability of h_1 making wrong prediction but h_2 making right prediction (blue prediction):

$$\mathbb{P}[h_1(\boldsymbol{x}_i) \neq h_2(\boldsymbol{x}_i) = f(\boldsymbol{x}_i)] = \frac{1}{2}$$

To make $E_{\text{in}}(h_1) = E_{\text{in}}(h_2)$, the number of (2) and (3) must be equal. Hence,

$$\mathbb{P}[E_{\text{in}}(h_1) = E_{\text{in}}(h_2)] = \frac{5!}{5!}(\frac{3}{8})^5 + \frac{5!}{1!1!3!}(\frac{3}{8})^3(\frac{1}{8})(\frac{1}{2}) + \frac{5!}{2!2!1!}(\frac{3}{8})(\frac{1}{8})^2(\frac{1}{2})^2 = \frac{3843}{32765}.$$

13.

Since $h_i(\boldsymbol{x}) = -h_{d+i}(\boldsymbol{x}),\ \forall i \in \{1, ..., d\}$, $E_{\text{in}}(h_i) = 1 - E_{\text{in}}(h_{d+i})$ and $E_{\text{out}}(h_i) = 1 - E_{\text{out}}(h_{d+i})$. Therefore,

$$|E_{\text{in}}(h_i) - E_{\text{out}}(h_i)| = |(1 - E_{\text{in}}(h_i)) - (1 - E_{\text{out}}(h_i))| = |E_{\text{in}}(h_{d+i}) - E_{\text{out}}(h_{d+i})|.$$

That is, $\mathcal{D}$ is a bad for h_i if and only if $\mathcal{D}$ is bad for h_{d+i}. Thus, we should consider these two hypothesis as the same class in terms of $|E_{\text{out}}(h) - E_{\text{in}}(h)|$. Therefore, there are d classes and thus

$$\mathbb{P}[\text{BAD } \mathcal{D} \text{ for } \mathcal{H}] \leq d \cdot 2\exp(-2\epsilon^2 N).$$

Multiple-Bin Sampling

14.

First we list the colors of the faces for each dice:

dice\number	1	2	3	4	5	6
A	o	g	o	g	o	g
B	o	g	g	g	o	o
C	o	o	o	o	o	g
D	o	g	g	o	g	o

Randomly pick one dice from the bag, we can compute the probabilities below:

$\mathbb{P}[3 \text{ is green}] = \frac{1}{2}$

$\mathbb{P}[1 \text{ is green}] = 0$

$\mathbb{P}[2 \text{ is green}] = \frac{3}{4}$

$\mathbb{P}[2 \text{ is orange}] = \frac{1}{4}$

$\mathbb{P}[4 \text{ is green}] = \frac{1}{2}$

$\mathbb{P}[5 \text{ is green}] = \frac{1}{4}$

Therefore, we choose (d).

15.

We denote the number of the dice A, B, C and D being drawn by four integers (between 0 and 5) x_1. x_2, x_3 and x_4, respectively.

Case(1) $x_3 = 0$ (C is not drawn)
This means that 2 is purely green. So the condition holds. The probability of this case is $\frac{3^5}{4^5}$.

Case(2) $x_3 > 1$
If $x_3 > 1$, then 1, 2, 3, 4 and 5 is impossible to be purely green. To make 6 purely green, B and D can't be drawn from the bag. So the probability is

$$\underbrace{(\frac{2}{4})^5}_{\text{B and D aren't drawn}} - \underbrace{(\frac{1}{4})^5}_{\text{B, C and D aren't drawn}}.$$

Since these two cases are exclusive, the answer is the sum of probability of these two cases (i.e. $\frac{274}{1024}$).

Appendix

Codes for 9.

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
import matplotlib.patches as mpatches

X = [[1, 0], [3, 2], [0, 2], [2, 3], [2, 4], [3, 5]]
X = np.array(X)
y = np.array([1, 1, 1, 0, 0, 0])

fig, ax = plt.subplots()
ax.plot(X[y==0, 0], X[y==0, 1], '.', marker='x')
ax.plot(X[y==1, 0], X[y==1, 1], '.', marker='o')
for i, x in enumerate(X):
    ax.text(x[0] + 0.1, x[1] + 0.2, r'$x_{}$'.format(i))
ax.axis([-1, 4, -1, 6])
plt.show()

def plot_perceptron(X, y, weights):
    w0, w1, w2 = weights[0], weights[1], weights[2]
    def perceptron_predict(x, y):
        return np.sign(w0 + w1 * x + w2 * y)
    fig, ax = plt.subplots()
    custom_cmap = ListedColormap(['#B2FF00','gold'])
    y_is_negative_patch = mpatches.Patch(color='#B2FF00', label=r'$\hat y$ = -1')
    y_is_positive_patch = mpatches.Patch(color='gold', label=r'$\hat y$ = +1')
    # coloring
    xs = np.linspace(-1, 4, 100)
    ys = np.linspace(-1, 6, 1000)
    xs, ys = np.meshgrid(xs, ys)
    ax.contourf(xs, ys, perceptron_predict(xs, ys), cmap=custom_cmap)

    # plot the margin
    if w2 != 0:
        x_s = np.linspace(-1, 4, 100)
        y_s = -(w0 + w1 * x_s) / w2
        ax.plot(x_s, y_s, 'b--')
    elif w1 != 0:
        y_s = np.linspace(-1, 6, 100)
        x_s = np.zeros(100) + (-w0) / w1
        ax.plot(x_s, y_s, 'b--')

    # marking data
    x_mark, = ax.plot(X[y==0, 0], X[y==0, 1], '.', marker='x', label=r'$y_n = -1$')
    o_mark, = ax.plot(X[y==1, 0], X[y==1, 1], '.', marker='o', label=r'$y_n = +1$')
    for i, x in enumerate(X):
        ax.text(x[0] + 0.1, x[1] + 0.2, r'$x_{}$'.format(i))

    ax.legend(handles=[y_is_positive_patch, y_is_negative_patch, o_mark, x_mark], framealpha=1)
    ax.axis([-1, 4, -1, 6])
    plt.show()
    return

# case1: E_ots = 0
plot_perceptron(X, y, [2.5, 0.15, -1])

# case2: E_ots = 1
plot_perceptron(X, y, [0.2, 0.15, 1])
```