Learning From Data Problems: Chapter I

J. David Giese

Exercise 1.4

See `exercise1_4.py` for code. Usually the algorithm takes 20 - 40 iterations to run, and matches the result fairly well. Of course, it depends on the training dataset.

Exercise 1.5

a) Learning
b) Design
c) Learning
d) Design
e) Design - I am not certain. The road system is interconnected, so gathering enough data for all the possible situations may be unfeasible. A simulation with parameters extracted from data may be more useful.

Exercise 1.6

a) Reinforcement learning, or supervised learning. In this example, what is the input space and output space? What training samples do we have? Assuming the input space is the set of all books, and the output space is whether a *particular person* will buy that book (i.e. buy, don't buy), then one may view this as supervised learning. However, just knowing the title of the book is probably not enough information.

A more useful input space would contain details about a particular person's book preferences and even their current mood, and the output space would be, for example, 5 books we should recommend. Unlike our first approach, this target function can be used on more than one person. In this case, our training data is likely of the form:

$$(\text{individual's buying history and mood}, \text{suggested book}, \text{buy})$$

Thus, this is clearly a reinforcement learning problem.

b) Reinforcement learning. The input space would be a particular tic-tack-toe situation (i.e. for each of the 9 squares, whether it was "x", "o", or "blank"). The output space would be which move should be chosen by the current player. We would reinforce

moves that ultimately led to the person winning the game.

c) This could be supervised learning OR unsupervised learning, depending on the training samples we had available. As supervised learning, the output space would be the set of all movie categories, and our training samples would have the movie (or its mathematically reduced description) as the input value and the category as the output value. As unsupervised learning, we would only know the movie (or its mathematically reduced description).

d) Reinforcement learning. The input space would, perhaps, be a melody and some parameters describing the style of music desired. The output would be the produced music. We would reinforce on how "good" listeners thought think the music is.

e) Reinforcement learning. The input space is details about the persons financial history. The output space is the maximum credit. We would reinforce on how much money the bank made (or lost) on the person.

Problem 1.1

$$\begin{aligned}\mathbb{P}(\text{2nd B}) &= \mathbb{P}(\text{BB Bag}) \cdot \mathbb{P}(\text{2nd B}|\text{BB Bag}) + \mathbb{P}(\text{BW Bag}) \cdot \mathbb{P}(\text{2nd B}|\text{BW Bag}) \\ &= \frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 0 = \frac{1}{2}\end{aligned}$$

Problem 1.2

a) The $+1$ and -1 regions are separated by the line where

$$\mathbf{w}^T\mathbf{x} = w_0 + w_1x_1 + w_2x_2 = 0$$

this can be rearranged so that

$$x_2 = -\frac{w_0}{w_2} - \frac{w_1}{w_2}x_1$$

thus we see $a = -\frac{w_0}{w_2}$ and $b = -\frac{w_1}{w_2}$. The last important detail is which side is the $+1$ region and which is the -1 region. The x-intercept of the dividing line makes a good reference point. By looking at a point just to the right of the x-intercept, $x_+ = (1, -\frac{w_0}{w_1} + \delta, 0)$ we see $h(\mathbf{x}_+) = \text{sign}(\mathbf{w}^T\mathbf{x}_+) = \text{sign}(\delta \cdot w_1)$. Thus if $w_1 > 0$ the $+$ region is the right, and vice versa.

b) The two cases are identical, except that the $+$ region is to the right for the first one, and to the left for the second.

Note that the there is a free "magnitude" inherent in $\mathbf{w}$. Two numbers are required to

specify the dividing line, and a "sign" is required to indicate which side is the positive side and which is the negative, thus there is a free "magnitude" left from the third number. For example $\mathbf{w} = (-1, 1, 0)$ is equivalent to $\mathbf{w} = (-2, 2, 0)$, but not equivalent to $\mathbf{w} = (1, -1, 0)$. Interpreting problem 1.3 would be easier if a constraint where placed on $\mathbf{w}$ to ensure a canonical form. I propose requiring $w_1^2 + ... + w_d^2 = 1$, which would make w_0 a proper threshold, and the remaining portion of $\mathbf{w}$ a unit vector. This canonical form simplifies the interpretation of the limits in problem 3 properly.

Problem 1.3

a) Since $\mathbf{w}^*$ separates the data, we know $h(\mathbf{x}) = \text{sign}(\mathbf{w}^{*T}\mathbf{x}_n) = y_n \; \forall n$, thus we have

$$y_n(\mathbf{w}^{*T}\mathbf{x}_n) = y_n(\text{sign}(\mathbf{w}^{*T}\mathbf{x}_n)|\mathbf{w}^{*T}\mathbf{x}_n|) = y_n^2|\mathbf{w}^{*T}\mathbf{x}| = |\mathbf{w}^{*T}\mathbf{x}_n| \geq 0$$

thus we have $\rho \geq 0$. If one further assumes that no data points fall on the dividing line, we have $\rho > 0$.

b) I'm not sure why they need the first equation here, as it isn't necessary to prove the second part of the problem, but for thoroughness we have from (a)

$$\begin{aligned}
y(t)(\mathbf{x}^T(t)\mathbf{w}^*) \geq \min y(t)(\mathbf{w}^{*T}\mathbf{x}(t)) > \rho \; \forall t \quad &\implies \\
y(t-1)(\mathbf{x}^T(t-1)\mathbf{w}^*) > \rho \quad &\implies \\
\mathbf{w}^T(t-1)\mathbf{w}^* + y(t-1)(\mathbf{x}^T(t-1)\mathbf{w}^*) > \mathbf{w}^T(t-1)\mathbf{w}^* + \rho \quad &\implies \\
(\mathbf{w}^T(t-1) + y(t-1)\mathbf{x}(t-1))\mathbf{w}^* > \mathbf{w}^T(t-1)\mathbf{w}^* + \rho \quad &\implies \\
\mathbf{w}^T(t)\mathbf{w}^* > \mathbf{w}^T(t-1)\mathbf{w}^* + \rho.&
\end{aligned}$$

Now for the interesting part: let $S(t)$ be the claim that $\mathbf{w}^T\mathbf{w}^* \geq t\rho$. We know $S(0)$ is true because $\mathbf{w}(0) = \mathbf{0}$. We assume $S(t)$, and have

$$\begin{aligned}
\mathbf{w}^T(t)\mathbf{w}^* > t\rho \quad &\implies \\
(\mathbf{w}^T(t) + y(t)\mathbf{x}^T(t))\mathbf{w}^* > t\rho + y(t)\mathbf{x}^T(t)\mathbf{w}^* \quad &\implies \\
\mathbf{w}^T(t+1)\mathbf{w}^* > t\rho + y(t)\mathbf{x}^T(t)\mathbf{w}^* \quad &\implies \\
\mathbf{w}^T(t+1)\mathbf{w}^* > (t+1)\rho,&
\end{aligned}$$

Thus by induction we have $S(t)$ to be true for all t.

Note that it would seem $\mathbf{w}$ diverges, however eventually you will run out of bad data points if the data is linearly separable.

c) Using the fact that $y(t-1)\mathbf{w}^T(t-1)\mathbf{x}(t-1) < 0$ because $y(t-1)$ and $\mathbf{w}^T(t-1)x(t-1)$ have opposite signs, we have

$$
\begin{aligned}
||\mathbf{w}(t)||^2 &= ||\mathbf{w}(t-1)||^2 + y(t-1)^2||\mathbf{x}(t-1)||^2 + 2y(t-1)\mathbf{w}^T(t-1)\mathbf{x}(t-1) \\
&< ||\mathbf{w}(t-1)||^2 + ||\mathbf{x}(t-1)||^2.
\end{aligned}
$$

d) Let $S(t)$ be the claim that $||\mathbf{w}||^2 \leq tR^2$. We know $S(0)$ is true because $\mathbf{w}(0) = \mathbf{0}$. We assume $S(t)$, and have

$$
\begin{aligned}
||\mathbf{w}(t)||^2 &\leq tR^2 \quad \Longrightarrow \\
||\mathbf{w}(t)||^2 + ||\mathbf{x}(t)||^2 &\leq tR^2 + ||\mathbf{x}(t)||^2 \quad \Longrightarrow \\
||\mathbf{w}(t+1)||^2 &\leq tR^2 + ||\mathbf{x}(t)||^2 \leq tR^2 + R^2 \quad \Longrightarrow \\
||\mathbf{w}(t+1)||^2 &\leq (t+1)R^2,
\end{aligned}
$$

thus we see $S(t) \Rightarrow S(t+1)$, and thus $S(t)$ is true for all t by induction.

e) Using the results from (b) and (d) we have

$$
\begin{aligned}
\mathbf{w}^T(t)\mathbf{w}^* &\geq t\rho \quad \Longrightarrow \\
1 &\geq \frac{\mathbf{w}^T(t)\mathbf{w}^*}{||\mathbf{w}^T(t)||||\mathbf{w}^*||} \geq \frac{t\rho}{||\mathbf{w}^T(t)||||\mathbf{w}^*||} \quad \Longrightarrow \\
tR^2 &\geq ||\mathbf{w}(t)||^2 \geq \frac{t^2\rho^2}{||\mathbf{w}^*||^2} \quad \Longrightarrow \\
t &\leq \frac{||\mathbf{w}^*||^2R^2}{\rho^2}.
\end{aligned}
$$

Note that the right hand side of the final inequality has two unknown quantities, $||\mathbf{w}^*||$ and ρ. As mentioned earlier, ρ, and thus also $||\mathbf{w}^*||$, have can scale arbitrarily which means the final values may depend on the arbitrary order that the miscategorized points were selected. Although the scaling will cancel out in the end, by using the canonical form suggested earlier, $\mathbf{w}$ will be more or less the same after each run.

It also makes the meaning of ρ consistent: the minimum normalized distance to the boundary. This makes it possible to estimate ρ. It may also be possible to estimate an upper bound on $||\mathbf{w}||$ using R, such that we could design an algorithm that would intelligently stop iteration.

Problem 1.4

For all of the following plots, the gray line is the hypothesis, h, the solid line is the target function, f, and the dots are the data set. I couldn't figure out how to create legends, otherwise I would have.

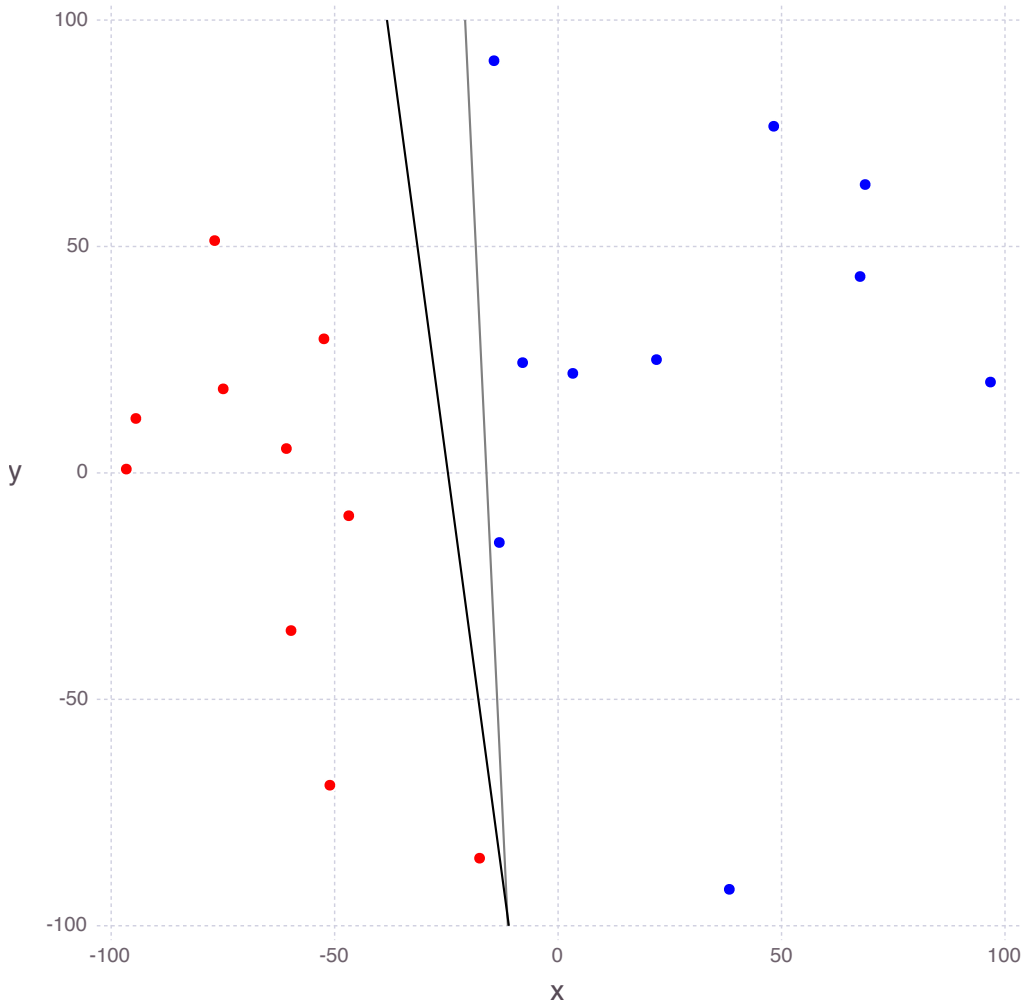

Figure 1: The solution for problem 1.4 b). The algorithm took 1,608 iterations to find a valid hypothesis. The final hypothesis and the target function are not as good of a match as they could be. This has to do with the implicit error measure that is used in the current perceptron algorithm; because we require that the data be linearly separable, our error measure is "the first hypothesis that fits". This is a poor error measure because it is unrealistic, non-deterministic, and very loose. It is unrealistic because noise in real datasets could lead to data sets with no match. It is non-deterministic because there are an infinite number of equally good hypothesis (any hypothesis that fits is equally good), and the PLA will pick a hypothesis depending on the order that mis-miscategorized points are iterated over. Finally, the hypotheses that are chosen are usually just barely correct and hug the current data. For example, in this case the hypothesis mis-categorizes a large area unnecessarily. This could be improved upon by using least-squares or some other error measure.

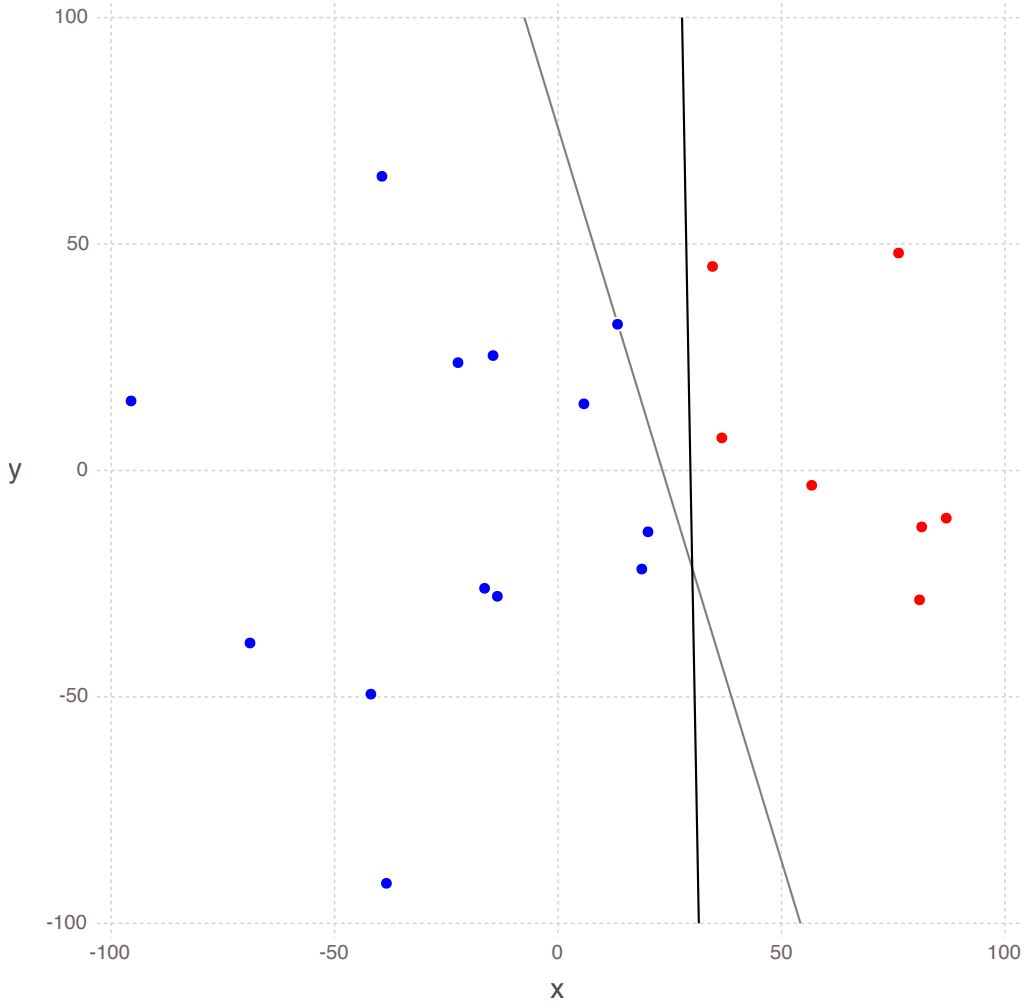

Figure 2: The solution for problem 1.4 c). The algorithm took 184 iterations to find a valid hypothesis.

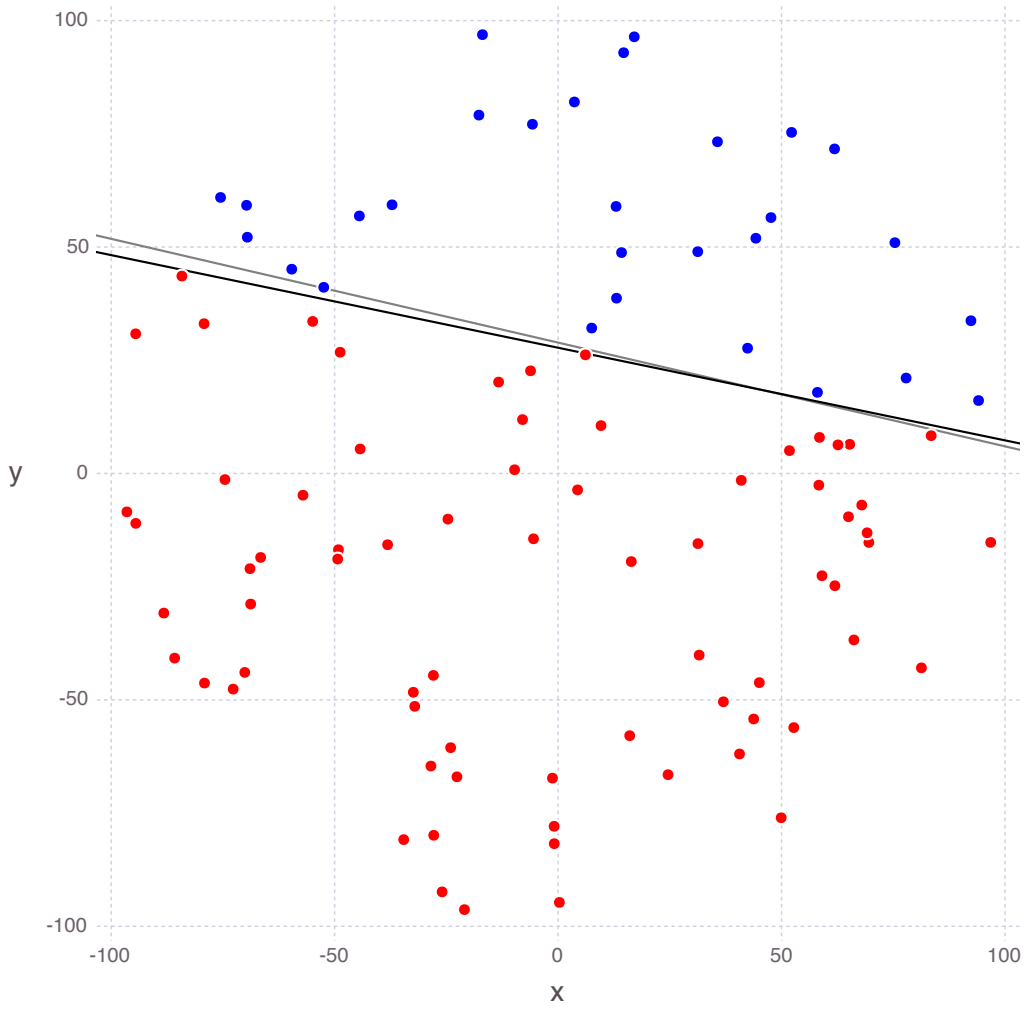

Figure 3: The solution for problem 1.4 d). The algorithm took 34,803 iterations to find a valid hypothesis.

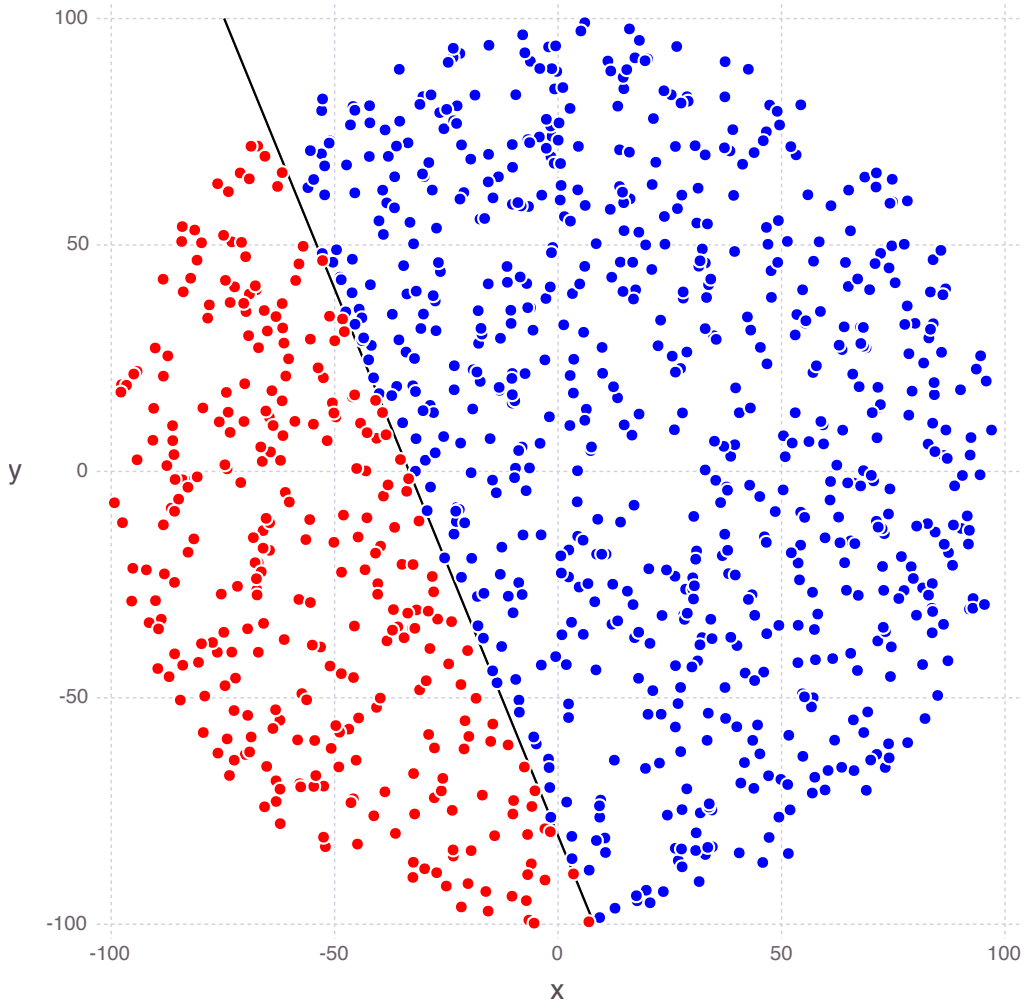

Figure 4: The solution for problem 1.4 e). The algorithm took 1,359,257 iterations to find a valid hypothesis. Note that the larger the data density the longer the algorithm takes to converge, but the more accurate the response is.

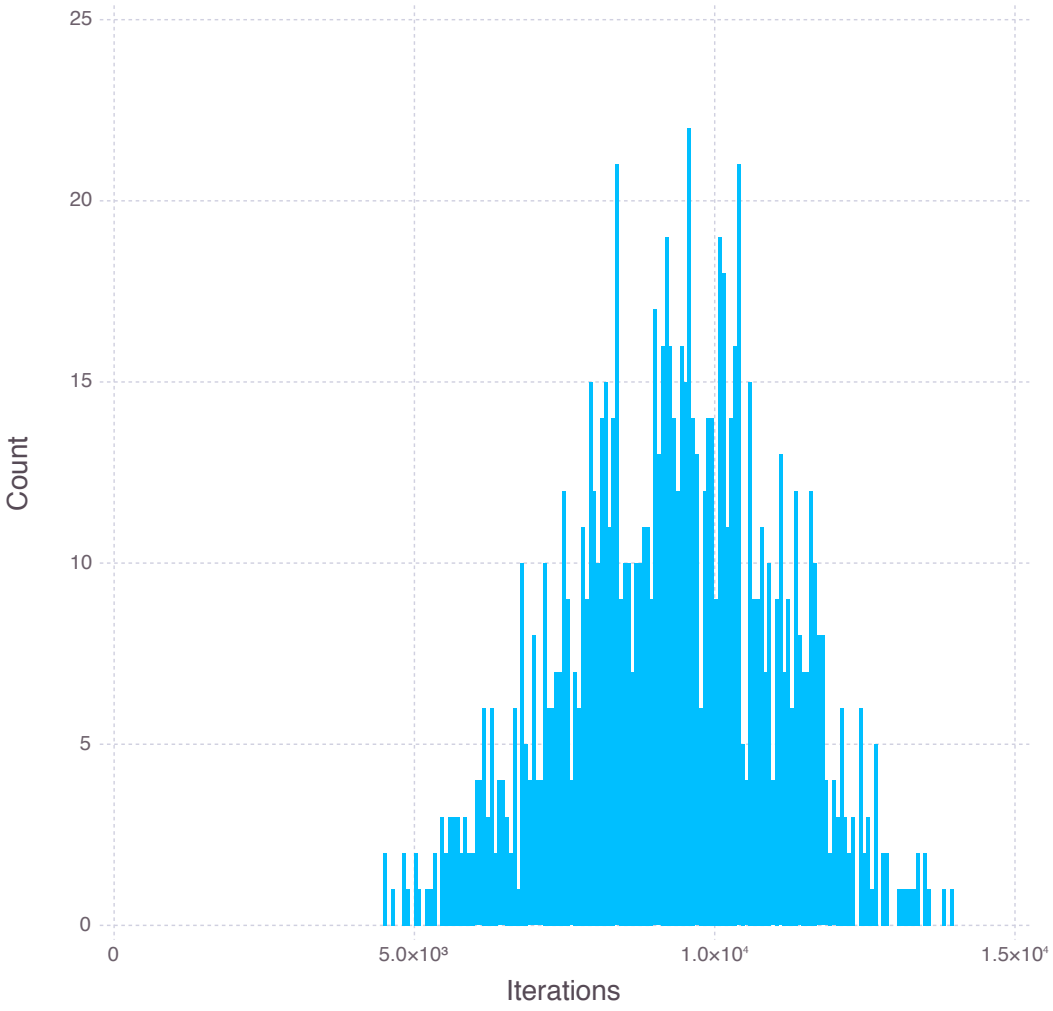

Figure 5: The solution for problem 1.4 f) and g).

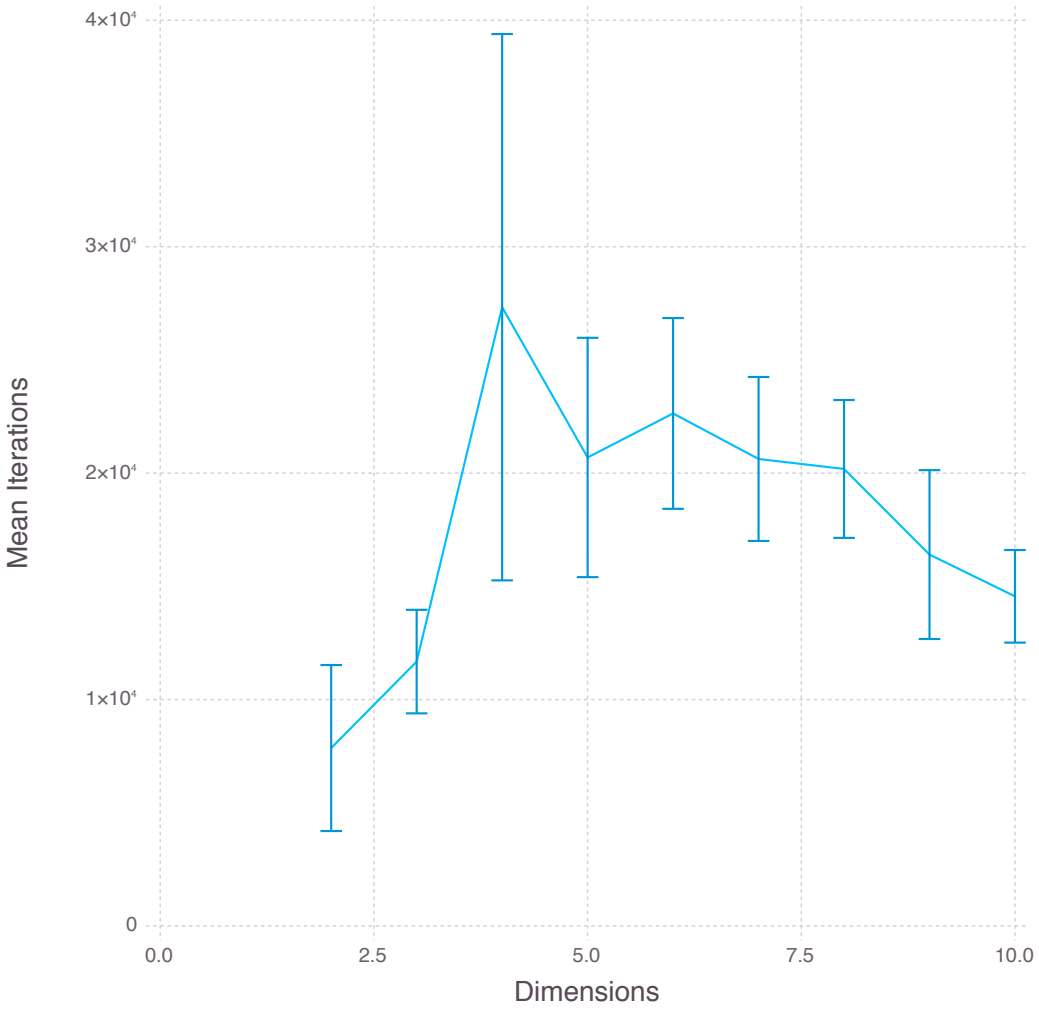

Figure 6: The runtime does not appear to increase as the dimensionality increases. The runtime was averaged over 100 runs (using 100 data points) for each dimension. Intuitively this makes sense, as each dimension is updated in parallel, furthermore there is nothing in the convergence bound proved in problem 1.3 that implies dimensionality would matter. The variability appears to decrease as the number of dimensions increases. This is likely because R will increase as the number of dimensions increases, because most of the points will exist near the edges of the n-sphere.

Note that the choice of f impacts the convergence speed a lot. Thus if f is being randomly chosen, the approach that it is generated could potentially alter some of the trends investigated in this problem. For example, if you pick your plane by selecting a random point in your n-sphere and using its direction as the normal and its magnitude as the offset, then as n increases the likelihood of the plane cutting through the edge of the n-sphere increases, and in the extreme case all of the points may be of a single type. On the other hand, if f is chosen to slice near the center of the n-sphere, then this won't be the case.

Problem 1.6

a) $\mathbb{P}_{\text{binomial}}(\nu = 0 | N = 10, \mu) = \binom{10}{0} \mu^0 (1-\mu)^{10-0} = (1-\mu)^{10}$.

b, c) The probability that at least one of M samples of 10 has $\nu = 0$ is equivalent to one less the probability of all of them not having $\nu = 0$. Let $S_i = \mathbb{P}(\text{Sample } i \text{ has } \nu \neq 0)$.

$$1 - \prod_{i=1}^{M} S_i = 1 - (1 - \mathbb{P}_{\text{binomial}}(\nu = 0 | N = 10, \mu))^M$$

Here are the numerical values to five places:

	$\mu = 0.05$	$\mu = 0.5$	$\mu = 0.8$
(a)	0.59874	0.00098	0.00000
(b)	1.00000	0.62358	1.00010
(c)	1.00000	1.00000	0.09733

Problem 1.8

a)

$$\begin{aligned}
\mathbb{P}[t \geq \alpha] &= \int_{\alpha}^{\infty} p_t(T) dT \\
&\leq \int_{\alpha}^{\infty} p_t(T) \frac{T}{\alpha} dT \\
&\leq \int_{\alpha}^{\infty} p_t(T) \frac{T}{\alpha} dT + \int_{0}^{\alpha} p_t(T) \frac{T}{\alpha} dT \\
&= \mathbb{E}[t]/\alpha.
\end{aligned}$$

Why not use $<$ for the first inequality? Given that $T/\alpha \leq 1$ within the integral, this may appear justified, however if $p_t(T) = \delta(T-1)$ and $\alpha = \lim_{\epsilon \to 0} 1 - \epsilon$, then all of the probability density is centered where $T/\alpha = 1$; this case is why the first equality is less than *or equal* to.

b) The random variable $(u-\mu)^2$ is non-negative, and $\mathbb{E}[(u-\mu)^2] \equiv \sigma^2$, thus using (a) we have $\mathbb{P}[(u-\mu)^2 \geq \alpha] \leq \frac{\sigma^2}{\alpha}$.

c) The expected value operator is linear, thus $\mathbb{E}[u] = \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}[u_n] = \mu$. Given that the set of u_n are iid, the variance is linear $\mathrm{Var}[u] = \frac{1}{N^2}\sum_{n=1}^{N}\mathrm{Var}[u_n] = \sigma^2/N$. Using the same logic as in (b) we have the solution.

Problem 1.9

a) Similarly to the first part of the previous problem

$$\begin{aligned}
\mathbb{P}[t \geq \alpha] &= \int_{\alpha}^{\infty} p_t(T)dT \\
&\leq \int_{\alpha}^{\infty} p_t(T)e^{s(t-\alpha)}dT \\
&\leq \int_{\alpha}^{\infty} p_t(T)e^{s(t-\alpha)}dT + \int_{-\infty}^{\alpha} p_t(T)e^{s(t-\alpha)}dT \\
&= e^{-s\alpha}\int_{-\infty}^{\infty} e^{st}p_t(T)dT \\
&= e^{-s\alpha}T(s).
\end{aligned}$$

b) A basic property of moment generating functions (e.g. $T(s)$) is that the m.g.f. of a sum is the product of their m.g.f.s, hence we have $\mathbb{E}_u(\exp(su)) = \mathbb{E}_u(\exp(\frac{s}{N}(u_1+u_2+ \ldots + u_N)) = U(\frac{s}{N})^N$, thus we have:

$$\mathbb{P}[u \geq \alpha] \leq (e^{-s\alpha}U(s))^N.$$

c) For a fair coin, we have $p(t) = \frac{1}{2}(\delta(t)+\delta(t-1))$, hence $U(s) = \frac{1}{2}(1+e^s)$. By inspection we see that $y(s) \equiv e^{-s\alpha}U(s)$ has no global maximum, and a single global minimum so long as $0 < \alpha < 1$, hence to minimize y we take its derivative with respect to s, equate it to 0, and solve for $s_{\min}$.

$$\begin{aligned}
\frac{d}{ds}y(s) = \frac{1}{2}[(-\alpha)e^{-\alpha s} + (1-\alpha)e^{s(1-\alpha)}] \quad &\Longrightarrow \\
0 = (-\alpha)e^{-\alpha s_{\min}} + (1-\alpha)e^{s_{\min}(1-\alpha)} \quad &\Longrightarrow \\
0 = (-\alpha) + e^{s_{\min}} - \alpha e^{s_{\min}} \quad &\Longrightarrow \\
s_{\min} = \ln\left(\frac{\alpha}{1-\alpha}\right).&
\end{aligned}$$

Note we care about $s_{\min}$ because it defines the upper bound for $\mathbb{P}[u_n \geq \alpha]$ (s is a free parameter on the rhs of the inequality in part (b)), after substituting $s_{\min}$ into $y(s)$ we have:

$$\mathbb{P}[u_n \geq \alpha] \leq \frac{(1-\alpha)^{\alpha-1}}{2\alpha^{\alpha}}.$$

d) Because $y(s) > 0$ for all s, the minimum of $y(s)^N$ occurs at the same location. Hence using the results of (b) and (c) we have

$$\mathbb{P}[u \geq \alpha] \leq \frac{(1-\alpha)^{\alpha-1\,N}}{2\alpha^{\alpha}} \quad .$$

This can be converted to the form in the textbook after a bunch of mathematical juggling that I don't have time to write out.

Problem 1.10

a)

$$E_{\text{off}} = \begin{cases} \frac{1}{2} & \text{if } M \text{ is even,} \\ \frac{1}{2}\frac{M-1}{M} & \text{if } M \text{ is odd and } N \text{ is even,} \\ \frac{1}{2}\frac{M+1}{M} & \text{if } M \text{ is odd and } N \text{ is odd.} \end{cases}$$

b) There are 2^M possible target functions that can generate $\mathcal{D}$.

c) If $E_{\text{off}} = \frac{k}{M}$ then there are k mismatches in the M remaining $\mathbf{x}$, hence there are $\frac{M!}{k!(M-k)!}$ target functions f that produce the specified off-training set error.

d) The probability of having $E_{\text{off}} = \frac{k}{M}$ is the result of part (b) divided by the result of part (c), hence we have

$$\begin{aligned} \mathbb{E}_f[E_{\text{off}}(h,f)] &= \sum_{k=0}^{M} \frac{k}{M} \frac{M!}{k!(M-k)!} \frac{1}{2^M} \\ &= \frac{1}{2^M} \sum_{k=0}^{M} \frac{(M-1)!}{(k-1)!((M-1)-(k-1))!} \\ &= \frac{1}{2^M} \sum_{k=0}^{M-1} \frac{(M-1)!}{k!((M-1)-k)!} \\ &= \frac{1}{2}. \end{aligned}$$

e) Because we are only considering the off-training-set error, whether $A_1(\mathcal{D})$ generates $\mathcal{D}$ or not, the result from (d) should stand equally well for any hypothesis.

Problem 1.11

$$E_{\text{in}}(h,f) = \frac{1}{N} \sum_{n=1}^{N} e(h(\mathbf{x}_n), y_n))$$

where e is defined according to the risk matrices. For example, the CIA pointwise error measure would be

$$e(h(\mathbf{x}_n), y_n) = \begin{cases} 1000 & \text{if } h(\mathbf{x}_n) = 1 \text{ and } y_n = -1, \\ 1 & \text{if } h(\mathbf{x}_n) = -1 \text{ and } y_n = 1, \\ 0 & \text{otherwise.} \end{cases}$$

Problem 1.12

a) We minimize E_{in} by taking its derivative with respect to h and solving for h_{min} such that the result is 0. Note we are assuming there is no global maximum.

$$\frac{dE_{\text{in}}(h)}{dh} = 2\sum_{n=1}^{N}(h - y_n) = 2hN - 2\sum_{n=1}^{N} y_n \quad \Longrightarrow$$

$$h_{\text{min}} = \frac{1}{N}\sum_{n=1}^{N} y_n.$$

b) The method used in (a) doesn't work quite as well, because the derivative of the absolute value has a discontinuity. However, within the scope of this problem, it makes sense to define the derivative of this discontinuity to be 0. Hence we have

$$\frac{dE_{\text{in}}(h)}{dh} = \sum_{n=1, h!=y_n}^{N} \frac{h - y_n}{|h - y_n|}.$$

The summand is either 1 or -1, hence the value that minimizes the summation must be a value of h such that half the data points are at most h, and the other half are at least h—in other words, h is the median.

c) h_{median} will be unaffected (unless perhaps you have a very small dataset), while h_{mean} will diverge.