1. **The problems best suited for machine learning.**
   i. Classifying prime and non-prime numbers.
   Not suited, because there is an explicit rule for classifying prime and non-prime numbers.
   ii. Detecting potential fraud in credit card charges.
   Best suited, we could train the machine by using some histories of fraud in credit card to detect potential fraud.
   iii. Determining the time it would take a falling object to hit the ground.
   Not suited, because there are some physical formula for calculating the time it would take a falling object to hit the ground.
   iv. Determining the optimal cycle for traffic lights in a busy intersection.
   Suited, without explicit relation rules between cycle times and traffic eficiency, we could use ML to find better efficiency by adjusting the cycle for traffic lights.
   v. Determining the age at which a particular medical test is recommended.
   Suited, because we don't have explicit rule or formula for determining the age at which a particular medical test is recommended.
2. **Types of learning for playing chess.**
   Reinforcement learning. By winning and losing the game that machine played, we give them awards or penalties, making it able to improve its playing strategy.
3. **Types of learning for categorizing books.**
   Unsupervised learning. Because we don't have explicit definitions or rules for classifying books, thus we use clustering method on articles to find the category of the book with highest possibility.
4. **Types of learning for recognizing faces.**
   Supervised learning, Because we have the explicit answer that there are faces in the pictures.
5. **Types of learning for scheduling experiments on mice to evaluate the potential of cancer medicines.**
   Action learning. The machine can improve its hypothesis by asking us desired output.
6. Because the output of target function $f(x)$ is always +1 and the output of hypothesis is +1 when k is odd for every $x = x_k$, half of the data have different outputs from $f(x)$ and $g(x)$.
   $$\mathrm{E}_{OTS}(g, f) = \frac{1}{L}\left(\left\lceil \frac{N+L}{2} \right\rceil - \left\lceil \frac{N}{2} \right\rceil\right)$$
7. **For all possible f : X → Y, how many of them can generate D in a noiseless setting?**
   Because D has only N sets, and each set has two kinds of output, there 2^N are possible f (x).
8. For any A_t, $\mathrm{E}_f\{\mathrm{E}_{OTS}(A_t(D), f)\}$
   $$= \frac{1}{2^L}\sum_{k=1}^{2^L} \mathrm{E}_{OTS}(A_t(D), f)$$
   $$= \frac{1}{2^L}\sum_{k=1}^{2^L} \frac{1}{L}\sum_{l=1}^{L} [|g_t(x_{N+1}) \neq f_k(x_{N+1})|] = \frac{1}{2^L L}\left(\frac{2^L}{2} \times L\right) = \frac{1}{2}$$

9. $\mu = 0.5$, P($\nu = \mu$) ?

$$\binom{10}{5}\left(\tfrac{1}{2}\right)^{5}\left(\tfrac{1}{2}\right)^{5} \approx 0.24$$

10. $\mu = 0.8$, P($\nu = \mu$) ?

$$\binom{10}{8}\left(\tfrac{8}{10}\right)^{8}\left(\tfrac{2}{10}\right)^{2} \approx 0.3$$

11. $\mu = 0.8$, P($\nu \le 0.1$) ?

$$\binom{10}{1}\left(\tfrac{8}{10}\right)^{1}\left(\tfrac{2}{10}\right)^{9} + \binom{10}{0}\left(\tfrac{8}{10}\right)^{0}\left(\tfrac{2}{10}\right)^{10} \approx 4.1\times10^{-6}$$

12. $\mu = 0.8$, what is the bound given by Hoeffding's Inequality for the probability of $\nu \le 0.1$?

$$P\left[\,|\nu-\mu| \ge \varepsilon\,\right] < 2e^{-2\varepsilon 2N}$$

$$P\left[\,|\nu-\mu| \ge 0.7\,\right] < 2e^{-2\times0.49\times10} \approx 1.11\times10^{-4}$$

13. Probability of 5 green one's if we pick 5 dices.

$$\left(\tfrac{1}{4}+\tfrac{1}{4}\right)^{5} = \tfrac{1}{32}$$

14. Probability of some numbers that is purely green if we pick 5 dices.

Case 1: 5A, 5B, 5C, 5D

$$4\times\binom{5}{5}\left(\tfrac{1}{4}\right)^{5} = \tfrac{1}{256}$$

Case 2: 4A1C, 4A1D, 4B1C, 4B1D

$$4\times\binom{5}{4}\left(\tfrac{1}{4}\right)^{4}\left(\tfrac{1}{4}\right)^{1} = \tfrac{5}{256}$$

Case 3: A2C, 3A2D, 3B2C, 3B2D

$$4\times\binom{5}{3}\left(\tfrac{1}{4}\right)^{3}\left(\tfrac{1}{4}\right)^{2} = \tfrac{10}{256}$$

Case 4: 2A3C, 2A3D, 2B3C, 2B3D

$$4\times\binom{5}{2}\left(\tfrac{1}{4}\right)^{2}\left(\tfrac{1}{4}\right)^{3} = \tfrac{10}{256}$$

Case 5: 1A4C, 1A4D, 1B4C, 1B4D

$$4\times\binom{5}{1}\left(\tfrac{1}{4}\right)^{1}\left(\tfrac{1}{4}\right)^{4} = \tfrac{5}{256}$$

Result probability

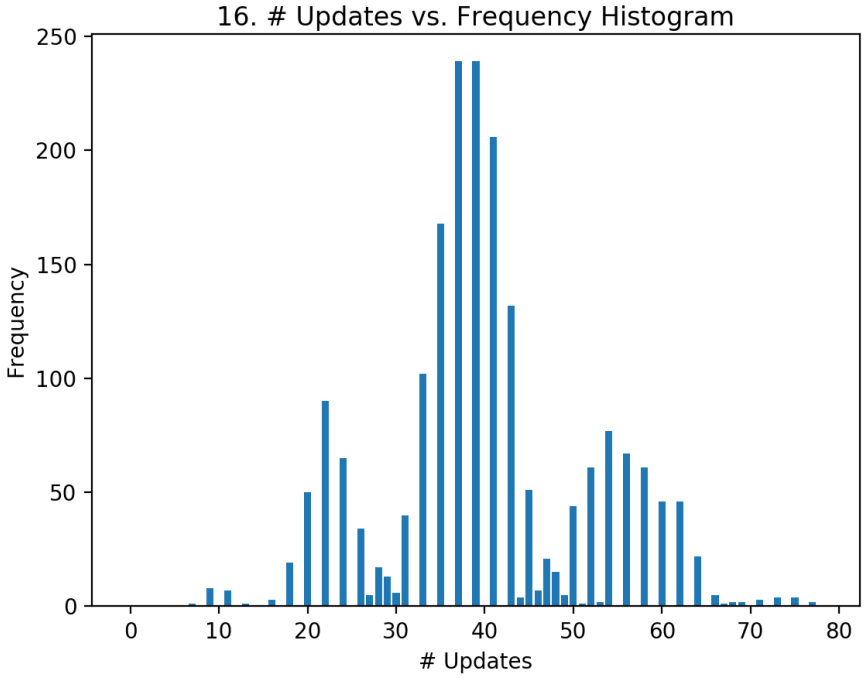
$$\tfrac{1}{256}+\tfrac{5}{256}+\tfrac{10}{256}+\tfrac{10}{256}+\tfrac{5}{256}=\tfrac{31}{256}$$

15. PLA on hw1_15_train.dat in order of input sequence

Update rule: $w_{i+1} <= w_i + y_{n(t)}x_{n(t)}$

Number of updates: 45

Data with most updates: index = 58, # update = 2



16. PLA on hw1_15_train.dat in order of fixed, pre-determined random cycles.

Update rule: $w_{i+1} <= w_i + y_{n(t)}x_{n(t)}$

Average number of updates: 40.2315

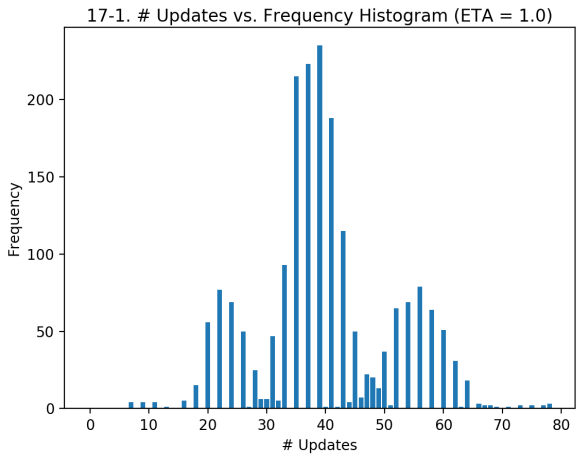
17. PLA on hw1_15_train.dat in order of fixed, pre-determined random 2000 cycles.

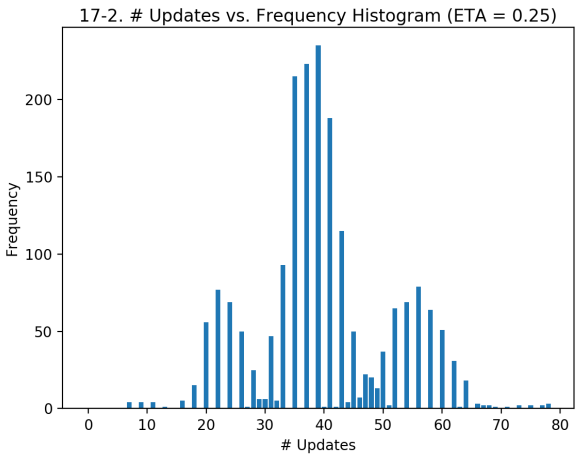
Original update rule: $w_{i+1} <= w_i + y_{n(t)}x_{n(t)}$

Average number of updates: 40

New update rule: $w_{i+1} <= w_i + \eta y_{n(t)}x_{n(t)}$, $\eta = 0.25$

Average number of updates: 40



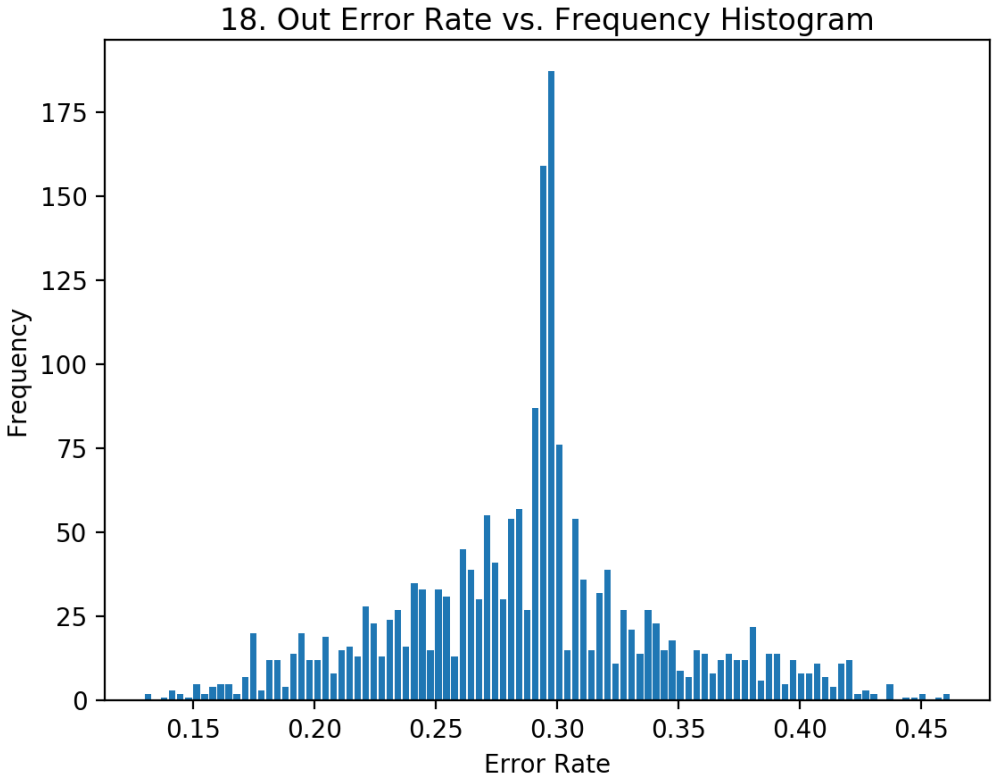


The result of two update rules are the same.

18. Pocket algorithm on hw1_18_train.dat in order of random 2000 cycles, 50 updates on D.

Average in error rate: 0.323673

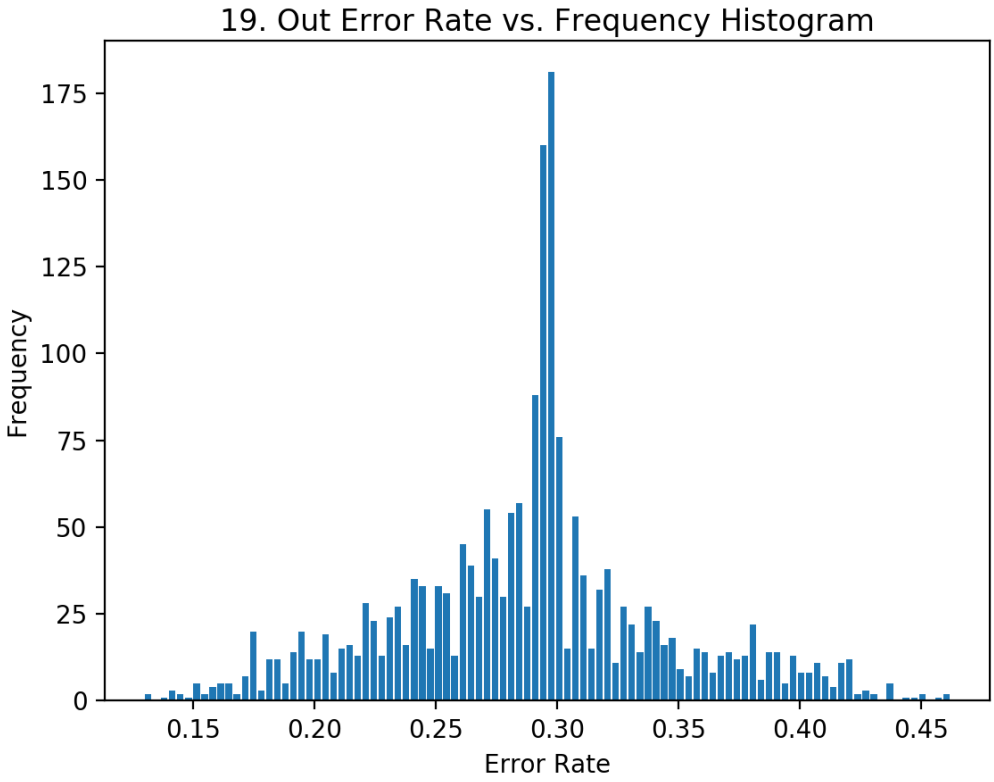
Average out error rate: 0.288769



19. Pocket algorithm on hw1_18_train.dat in order of random 2000 cycles, 100 updates on D.

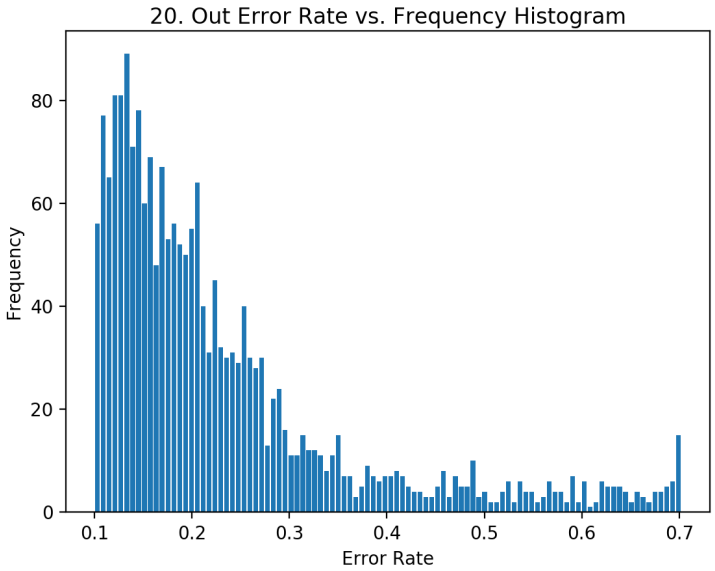
Average in error rate: 0.323422

Average out error rate: 0.288855



20. PLA on hw1_18_train.dat in order of fixed, pre-determined random 2000 cycles, 100 updates.

Average out error rate: 0.233642



The result is better than 19.

21. New update rule: $w_{t+1} = w_t + y_{n(t)}x_{n(t)}\left\lfloor \frac{-y_{n(t)}w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 1 \right\rfloor$

$$w_t + y_{n(t)}x_{n(t)}\left[\frac{-y_{n(t)}w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 1\right] \le w_{t+1} \le w_t + y_{n(t)}x_{n(t)}\left[\frac{-y_{n(t)}w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 2\right]$$

$$y_{n(t)}x_{n(t)} \le w_{t+1} \le 2y_{n(t)}x_{n(t)}$$

$$y_{n(t)}\|x_{n(t)}\|^2 \le w_{t+1}^T x_{n(t)} \le 2y_{n(t)}\|x_{n(t)}\|^2$$

By the inequation above, $\text{sign}\left(w_{t+1}^T x_{n(t)}\right) = \text{sign}(y_{n(t)})$

Thus, w_{t+1} always classifies $(x_{n(t)}, y_{n(t)})$.

22. No.

Assume that we will have a perfect w_f *s. t.* $y_n = \text{sign}\left(w_f^T x_n\right)$

$$w_f^T w_{t+1} = w_f^T\left(w_t + y_{n(t)}x_{n(t)}\left\lfloor \frac{-y_{n(t)}w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 1 \right\rfloor\right) = w_f^T w_t + w_f^T y_{n(t)}x_{n(t)}\left\lfloor \frac{-y_{n(t)}w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 1 \right\rfloor$$

$$w_f^T w_t + w_f^T y_{n(t)}x_{n(t)}\left[\frac{-y_{n(t)}w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 1\right] \le w_f^T w_{t+1} \le w_f^T w_t + w_f^T y_{n(t)}x_{n(t)}\left[\frac{-y_{n(t)}w_t^T x_{n(t)}}{\|x_{n(t)}\|^2} + 2\right]$$

$$w_f^T y_{n(t)}x_{n(t)} \le w_f^T w_{t+1} \le 2w_f^T y_{n(t)}x_{n(t)}$$

By the same method, we can approach:

$$w_f^T y_{n(t+1)}x_{n(t+1)} \le w_f^T w_{t+2} \le 2w_f^T y_{n(t+1)}x_{n(t+1)}$$

The w_t might not more aligned with w_f after an update, and the updated w_{t+1} only ensures that the point it referenced last time is perfectly seperated.