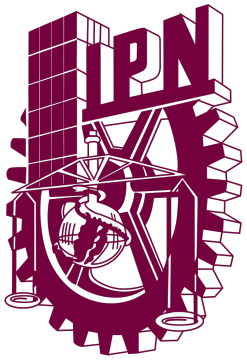

Instituto Politécnico Nacional

Centro de Investigación en cómputo

Chapter 1: The learning problem

Subject: Introducción a machine learning

Alumno:

- Carpintero Mendoza Marcos Mauricio

Profesor:
Menchaca Mendez Ricardo

14/10/19

Índice

1. Exercises.

1.1. Problem setup.

1.1.1. Components of learning.

Exercise 1.1 Express each of the following tasks in the framework of learning from data by specifying the input space $\mathcal{X}$, output space $\mathcal{Y}$, target function $f : \mathcal{X} \to \mathcal{Y}$, and the specifics of the data set that we will learn from.
Solution:

1. Medical diagnosis: A patient walks in with a medical history and some symptoms, and you want to identify the problem.

   Solution:

   - Let $\mathcal{X}$ be the following finite feature vector in which each element is a boolean variable:

   $$\mathcal{X} = \{\text{weight loss, tingling sensations, fatigue, red gums}\}$$

   - Let $\mathcal{Y}$ a binary variable:

   $$\mathcal{Y} = \{\text{diabetes, fever}\}$$

   - Suppose a target function, let $f : \mathcal{X} \to \mathcal{Y}$

   $$f = \begin{cases} \text{diabetes} & \text{if sum}(\mathcal{X}) \geqslant 3 \\ \text{fever} & \text{if sum}(\mathcal{X}) < 3 \end{cases}$$

   - Finally, the sample training data set:

   $$D = \{(\{0,0,0,0\}, fever), (\{1,0,0,0\}, fever), (\{0,1,0,0\}, fever) \\ , ..., (\{1,1,0,1\}, diabetes), (\{1,1,1,0\}, diabetes), (\{1,1,1,1\}, diabetes)\}$$

2. Handwritten digit recognition (for example postal zip code recognition for mail sorting)

   - Let $\mathcal{X}$ be the following finite feature vector in which each element is a boolean variable:

3. Determining if an email is spam or not.

   - Let $\mathcal{X}$ be the following finite vector in which each element is a typical spam word:
     Note1: The words in $\mathcal{X}$ were added after using stemming so that we avoid similar words.

   $$\mathcal{X} = \{discount, includ, cheap, free, cost, clerance, prize, low\}$$

     Note2: Keep in mind: $\forall x_i \in \mathcal{X} \exists k_i \in \mathbb{N} : k_i =$ number of x_i in the email.

   - Let $\mathcal{X}$ be the following finite vector.

   $$\mathcal{Y} = \{\text{Spam, Nonspam}\}$$

- Suppose the next target function $f : \mathcal{X} \rightarrow \mathcal{Y}$, such that:

$$f = \begin{cases} \text{Spam} & \text{if } \sum_{i=0}^{\text{len}(\mathcal{X})} x_i \geqslant 5 \\ \text{Nonspam} & \text{otherwise} \end{cases}$$

- Finally, the data training set:

$$D = \{(1,3,0,0,0,1,3,0,Spam), (\{1,1,1,1,0,2,0,0\},Spam), (\{1,0,0,0,0,0,1,0\},Nonspam), (\{0,1,0,1,0,1,0,0,0\},NonSpam), ..., (\{1,0,0,3,0,0,0,0\},Nonspam)\}$$

4. Predicting how an electrical load varies with the price, temperature and day of the week.

- Let X the feature vector which elements are the mentioned ones above.

$$\mathcal{X} = \{\text{Price, Temperature, Day}\}$$

- Let Y the vector with two elements:

$$\mathcal{Y} = \{\text{High, Low}\}$$

- Suppose the target function f as:

$$f = \begin{cases} \text{Low} & \text{if Day} \in \text{Weekday} \vee \text{Price} < \$15 \vee \text{Temperature} < 20C \\ \text{High} & Otherwise \end{cases}$$

- Finally, data training set:

$$D = \{(\{Monday, 20, 10\}, Low), (\{Friday, 10, 25\}, Low), ..., (\{Thursday, 20, 25\}, High), (\{Wednesday21, 30\}, High)\}$$

5. A problem of interest to you which there is not analytic solution, but you have data from which to construct an empirical solution.
Well, the situation is: Predicting the citizens salary base on each year of high education studied (i.e. The salary increases from the first year in degree).

- Let X feature vector which elements are the years studied starting from bachelor.

$$\mathcal{X} = \{\text{Bachelor, Master, PhD, Post doctoral}\}$$

Note: If x_{i+1} exists, then x_i must be completed.

- Let Y the vector which describe the salary range.

$$\mathcal{Y} = \{\text{Super-High, High, Normal}\}$$

- Suppose the function f as:

$$f = \begin{cases} \text{Super-high} & \text{if Post-Doctoral} \geqslant 1 \\ \text{High} & \text{if Master} \leqslant 2 \\ \text{Normal} & \text{if Bachelor} \leqslant 3 \end{cases}$$

- Finally, data training set:

$$D = \{(\{1,0,0,0\}, Normal), (\{5,1,0,0\}, High), ..., (\{5,3,2,0\}, Super - High), (\{5,3,3,1\}, Super - High)\}$$

1.1.2. A simple learning model.

Exercise 1.3 The weight update rule in (1.3) has the nice interpretation that it moves in the direction of classifying x(t) correctly.
Solution:

a) Show that $y(t)W^T(t)x(t) < 0$ [HINT: $x(t)$ is misclassified by $w(t)$]
First, by definition a correct classification:

$$\text{Sign}(W^T(t)x(t)) > 0 \quad \rightarrow \quad y(t) = 1$$

$$\text{Sign}(W^T(t)x(t)) \leqslant 0 \quad \rightarrow \quad y(t) = -1$$

Thus, misclassification means:

$$\text{Sign}(W^T(t)x(t)) > 0 \quad \rightarrow \quad y(t) = -1$$

$$\text{Sign}(W^T(t)x(t)) \leqslant 0 \quad \rightarrow \quad y(t) = 1$$

Finally, for both cases:

$$y(t)W^T(t)x(t) < 0$$

b) Show that $y(t)W^T(t+1)x(t) > y(t)W^T(t)x(t)$ [HINT: Use (1.3)]
Since the last proof we continue:

$$\begin{aligned} y(t)W^T(t+1)x(t) &= y(t)(W(t) + y(t)x(t))^T x(t) \\ &= y(t)W(t)^T x(t) + (y(t))^2 ||x(t)||^2 \\ &> y(t)W^T(t)x(t) \end{aligned}$$

c) As far as classifying $x(t)$ is concerned, argue that we move from $W(t)$ to $W(t+1)$ is a move in the right direction,
As we know, a correct classification (Check a)) and the definition of dot product:

$$W(t)^T x(t) = \| W^T \| \| x \| \cos(\alpha)$$

$$\cos(\alpha) = \frac{W(t)^T x(t)}{\| W^T \| \| x \|} \quad \rightarrow \quad \cos(\alpha) \propto W(t)^T x(t)$$

$$\cos(\alpha_{new}) \propto W(t+1)^T_{new} x(t)$$

$$\begin{aligned} \cos(\alpha_{new}) &\propto (W(t) + x(t))^T x(t) \\ &\propto W(t)^T x(t) + x(t)^T x(t) \\ &\propto \cos(\alpha) + x(t)^T x(t) \\ \cos(\alpha_{new}) &> \cos(\alpha) \end{aligned}$$

$$\begin{aligned} \cos(\alpha_{new}) &\propto (W(t) - x(t))^T x(t) \\ &\propto W(t)^T x(t) - x(t)^T x(t) \\ &\propto \cos(\alpha) - x(t)^T x(t) \\ \cos(\alpha_{new}) &< \cos(\alpha) \end{aligned}$$

Then, for each case shown above, we can identify that if we pick a misclassified point is because the $\cos\alpha$ gave a wrong value, so we must increase or decrease the value of α such makes that point a correct classification.

1.2. Is Learning Feasible?

1.2.1. Outside of the data set.

Exercise 1.7

x	y	g	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
0 0 0	○	○	○	○	○	○	○	○	○	○
0 0 1	●	●	●	●	●	●	●	●	●	●
0 1 0	●	●	●	●	●	●	●	●	●	●
0 1 1	○	○	○	○	○	○	○	○	○	○
1 0 0	●	●	●	●	●	●	●	●	●	●
1 0 1		?	○	○	○	○	●	●	●	●
1 1 0		?	○	○	●	●	○	○	●	●
1 1 1		?	○	●	○	●	○	●	○	●

For each of the following learning scenarios in the above problem, evaluate the performance of g on the three points in $\mathcal{X}$ outside $\mathcal{D}$. To measure the performance, compute how many of the 8 possible target functions agree with g on all three points, on two of them, on one of them , and on none of them.

Observation: Since there are three missing vectors in X, so there are eight missing functions. For each f_i we must analyze the output in it which is function of the X vector and apply the following formula to get the score:

Solution:

$$Score = \sum_{i=0}^{3} i(\#\text{ function agrees } i \text{ times})$$

a) $\mathcal{H}$ has only two hypotheses, one that always return • and one that always returns ○. The learning algorithm picks the hypotheses that match the data set most.

$$\#\text{ times functions always returns } 1 = 1$$

$$\#\text{ times functions once returns 1 and twice } 2 = 3$$

$$\#\text{ times functions once returns 2 ant twice } 1 = 3$$

$$\#\text{ times functions always returns } 0 = 1$$

$$Score = 0(1) + 1(3) + 2(3) + 3(1) = 9$$

b) The same $\mathcal{H}$, but the learning algorithm now picks the hypotheses that matches the *least*.

$$\#\text{ times functions always returns } 1 = 1$$

$$\#\text{ times functions once returns 1 and twice } 2 = 3$$

$$\#\text{ times functions once returns 2 ant twice } 1 = 3$$

$$\#\text{ times functions always returns } 0 = 1$$

$$Score = 0(1) + 1(3) + 2(3) + 3(1) = 9$$

c) $\mathcal{H} = \{XOR\}$ (only one hypothesis is always picked), where XOR is defined by $XOR(x) = \bullet$ if the number of 1's in $\mathcal{X}$ is odd and $XOR(x) = \circ$ if the number is even.

$$\#\text{ times functions agree 0 times} = 1$$
$$\#\text{ times functions agree 1 times} = 2$$
$$\#\text{ times functions agree 2 times} = 2$$
$$\#\text{ times functions agree 3 times} = 1$$
$$Score = 0(1) + 1(3) + 2(3) + 3(1) = 9$$

d) $\mathcal{H}$ contains all possible hypotheses (Boolean functions on threes variables), and the learning algorithm picks the hypotheses that agrees with all training examples, but otherwise disagrees the most with the XOR.

$$\#\text{ times functions agree 0 times} = 1$$
$$\#\text{ times functions agree 1 times} = 2$$
$$\#\text{ times functions agree 2 times} = 2$$
$$\#\text{ times functions agree 3 times} = 1$$
$$Score = 0(1) + 1(3) + 2(3) + 3(1) = 9$$

1.2.2. Probability to the rescue.

Exercise 1.8 If $\mu = 0.9$, what is the probability that a sample of 10 marbles will have $\upsilon \leqslant 0.1$ [HINTS: 1. Use the binomial distribution. 2. The answer is a very small number.]
Solution:
Let $R = \#$ red marbles & $G = \#$ green marbles

$$P(R) = 0.9 \quad \rightarrow \quad P(G) = 1 - P(R) = 0.1$$

In other hand:

$$P(\upsilon \leqslant 0.1) = P(\frac{R}{N} \leqslant 0.1) = P(R \leqslant 1)$$
$$R \sim B(10, 0.1)$$

Using the formula we get:

$$P(R \leqslant 1) = P(R = 0) + P(R = 1) = 0.1^{10} + 9 \times 10^{-9} = 10^{-9}(9 + 0.1) = 9.1 \times 10^{-9}$$

Exercise 1.9 If $\mu = 0.9$, use the Hoeffding Inequality to bound the probability that a sample fo 10 marbles will have $\mu \leqslant 0.1$ and compare the answer to the previous exercise.
Solution:

$$\upsilon - \mu = -0.8 \rightarrow |\upsilon - \mu| \geqslant 0.8$$
$$\text{If } \epsilon = 0.8 \quad \rightarrow \quad P(|\nu - \mu| > 0.8^{-}) \leqslant 2e^{-2\times 0.8^2 \times 10} \approx 5.52 \times 10^{-6}$$
$$P(\nu \leqslant 0.1) \leqslant P(|\nu - \mu| > 0.8^{-}) \Leftrightarrow 9.1 \times 10^{-9} \leqslant 5.52 \times 10^{-6}$$

Exercise 1.10 Here is an experiment that illustrates the difference between a single bin and multiple bins. Run a computer simulation for flipping 1, 000 fair coins. Flip each coin independently times. Let's focus on 3 coins as follows: c_1 is the first coin flipped; c_{rand} is a coin you choose at random; c_{min} is the coin that had the minimum frequency of heads (pick the earlier one in case of a tie). Let v_1 , v_{rand} and v_{min} be the fraction of heads you obtain for the respective three coins.

a) What is μ for the th ree coins selected?

b) Repeat this entire experiment a large n umber of times (e.g. , 100, 000 runs of the entire experiment) to get several instances of v_1 , v_{rand} and v_{min} and plot the histograms of the distributions of v_1 , v_{rand} and v_{min}· Notice that which coins end up being c_{rand} and c_{min} may differ from one run to another.

c) Using b), plot estimates for $P[|\nu - \mu| > \epsilon]$ as a function of ϵ, together with the Hoeffding bound $2e^{-2^2N}$ (on the same graph).

Solution:

a) All the coins are fair, so: $\mu = 0.5$

‘

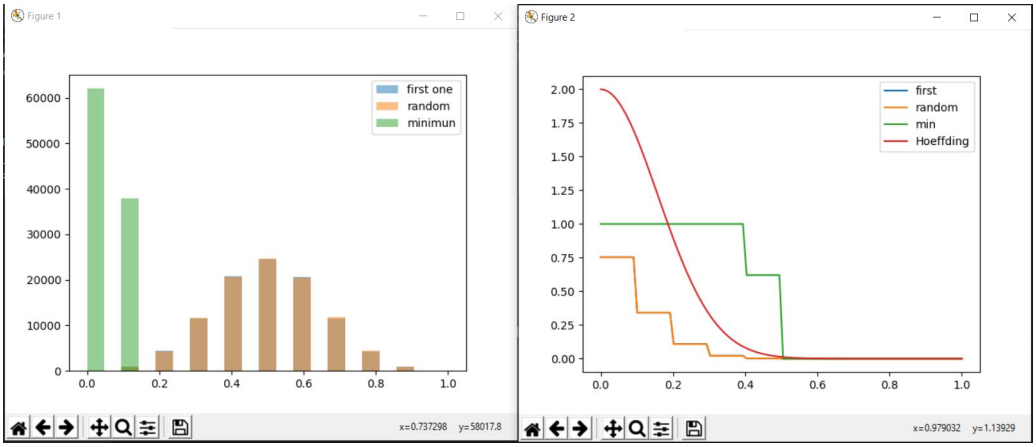

Figura 1: The histogram (left) shows the fraction of heads for the selected coins, and the plot (right) shows the probability and Hoeffding for the same ones coins.

Code:

```python
import numpy as np
import matplotlib.pyplot as plt

def simulate(ncoin, ntoss):
    outcome = np.random.choice((0., 1.), size=(ncoin, ntoss))
    heads = np.sum(outcome, axis=1)
    heads /= 10
    return heads[0], np.random.choice(heads), np.min(heads)

def plott1():
    bins = np.linspace(0, 1, 22)
    plt.hist(experiments[:, 0], bins, alpha=0.5, label="first one")
    plt.hist(experiments[:, 1], bins, alpha=0.5, label="random")
    plt.hist(experiments[:, 2], bins, alpha=0.5, label="minimun")
    plt.legend()
    plt.show()

def P(outcomes, mu, epsilon):
    less = np.sum(outcomes < mu - epsilon)
    greater = np.sum(outcomes > mu + epsilon)
    return (less + greater) / len(outcomes)

def hoeffding(epsilon, N):
    return 2 * np.exp(-2 * epsilon * epsilon * N)

def plott2():
    epsilons = np.linspace(0, 1, 100)
    P_first = [P(experiments[:, 0], mu, epsilon) for epsilon in epsilons]
    P_random = [P(experiments[:, 1], mu, epsilon) for epsilon in epsilons]
    P_min = [P(experiments[:, 2], mu, epsilon) for epsilon in epsilons]
    P_hoeffding = [hoeffding(epsilon, ntoss) for epsilon in epsilons]
    plt.plot(epsilons, P_first, label="first")
    plt.plot(epsilons, P_random, label="random")
    plt.plot(epsilons, P_min, label="min")
    plt.plot(epsilons, P_hoeffding, label="Hoeffding")
    plt.legend()
    plt.show()

if __name__ == '__main__':
    ncoin = 1000 # Number of fair coins for each experiment
    ntoss = 10 # Number of toss for each coin
    mu = 0.5 # Probability
    nexperiment = 100000 # Number of experiments
    experiments = np.empty((nexperiment, 3))
    for i in range(nexperiment):
        first, rand, minimun = simulate(ncoin, ntoss)
        experiments[i, 0], experiments[i, 1], experiments[i, 2] = first, rand, minimun
    plott1()
    plott2()
```

1.3. Error and noise

1.3.1. Noisy targets.

Exercise 1.13 Consider the bin model for a hypothesis h that makes an error with probability μ in approximating a deterministic target function (both h and are binary functions). If we use the same h to approximate a noisy version of f given by:

$$P(y \mid \mathbf{x}) = \begin{cases} \lambda & y = f(\mathbf{x}), \\ 1-\lambda & y \neq f(\mathbf{x}). \end{cases}$$

a) What is the probability of error that h makes in approximating y?

b) At what value of λ will the performance of h be independent of μ? [Hint: The noisy target will look completely random.]

Solution:

a) First, we just know that $\mathsf{P}[\mathsf{h}(\mathsf{x}) \neq \mathsf{f}(\mathsf{x})] = \mu$ and we want to know the $\mathsf{P}[\mathsf{h}(\mathsf{x}) \neq \mathsf{y}]$, so there are two cases:

1. $\mathsf{P}[\mathsf{h}(\mathsf{x}) \neq \mathsf{y}] = \mathsf{P}[\mathsf{h}(\mathsf{x}) = \mathsf{f}(\mathsf{x})]\mathsf{P}[\mathsf{f}(\mathsf{x}) \neq \mathsf{y}]$
2. $\mathsf{P}[\mathsf{h}(\mathsf{x}) \neq \mathsf{y}] = \mathsf{P}[\mathsf{h}(\mathsf{x}) \neq \mathsf{f}(\mathsf{x})]\mathsf{P}[\mathsf{f}(\mathsf{x}) = \mathsf{y}]$

If we sum the two parts we obtain the answer:

$$= (1-\mu)(1-\lambda) + (\mu)(\lambda)$$

$$= 1 + 2\mu\lambda - \mu - \lambda$$

b) When we say independent of μ it means that whichever its value could be does not matter in the original prediction: $h(x) = y$. So, in what we are really interested is in the value for λ, in the previous result, such that μ disappears:

$$2\mu\lambda - \mu = 0$$

$$2\mu\lambda = \mu$$

$$\lambda = \frac{1}{2}$$

2. Problems.

1.1) We have 2 opaque bags, each containing 2 balls. One bag has 2 black balls and the other has a black and a white ball. You pick a bag at random and then pick one of the balls in that bag at random. When you look at the ball it is black. You now pick the second ball from that same bag. What is the probability that this ball is also black?

Solution:

Let be, $\mathbf{B_k}$: k-Ball is black, so we want to know:

$$\mathbb{P}(B_2|B_1) = \frac{\mathbb{P}(B_1B_2)}{\mathbb{P}(B_1)}$$

$$= \frac{\sum_k^n \mathbb{P}(B_1B_2|Bag_k)\mathbb{P}(Bag_k)}{\sum_k^n \mathbb{P}(B_1|Bag_k)\mathbb{P}(Bag_k)}$$

$$= \frac{1(\frac{1}{2}) + \frac{1}{2}(0)}{1(\frac{1}{2}) + \frac{1}{2}(\frac{1}{2})}$$

$$= \frac{2}{3}$$

1.2) Consider the perceptron in two dimensions: $h(x) = sgn(\mathbf{w^T x})$ where $\mathbf{w} = [w_0, w_1, w_2]^T$ and $\mathbf{x} = [1, x_1, x_2]^T$. Technically, x has three coordinates, but we call this perceptron two-dimensional because the first coordinate is fixed at 1.

1) Show that the regions on the plane where $h(x) = +1$ and $h(x) = -1$ are separated by a line. If we express this line by the equation $x2 = ax1 + b$, what are the slope a and intercept b in terms of w_0, w_1, w_2?
2) Draw a picture for the cases $w = [1, 2, 3]$ and $w = -[1, 2, 3]^T$.

Solution:

1) We have the following perceptron: $h(x) = sgn(\mathbf{w^T x})$, so:

$$h(x) = +1 \quad \rightarrow \quad w^T x > 0$$

$$h(x) = -1 \quad \rightarrow \quad w^T x < 0$$

$$\Rightarrow w^T x = 0 \text{ is the separator line}$$

The equation is:

$$w_0 + w_1x_1 + w_2x_2 = 0$$

$$w_0 + w_1x_1 = -w_2x_2$$

$$\frac{w_0 + w_1x_1}{-w_2} = \frac{-w_2x_2}{-w_2}$$

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

The constants are next to the x_i, so:

$$b = -\frac{w_0}{w_2}; a = -\frac{w_1}{w_2}$$

2) We obtain the following plots for each vector:

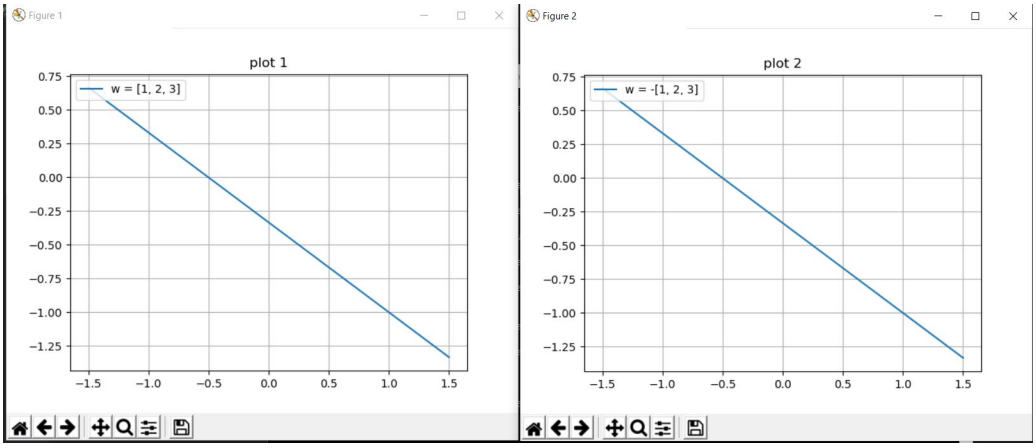

Figura 2: The separator line is identical in both plots, but the area in which h(x) = +1 puts each xi in the left is on top and in the right is on bottom.

Code:

```
import matplotlib.pyplot as plt
import numpy as np

if __name__ == '__main__':
    x = np.linspace(-1.5, 1.5, 80)
    w1 = [1, 2, 3]
    w2 = [-1, -2, -3]
    y1 = -(w1[0]/w1[2]) - (w1[1]*x)/w1[2]
    plot1 = plt.figure(1)
    plt.plot(x, y1, label='w = [1, 2, 3]')
    plt.title('plot 1')
    plt.legend(loc='upper left')
    plt.grid()
    plot1.show()

    y2 = -(w2[0]/w2[2]) - (w2[1]*x)/w2[2]
    plot2 = plt.figure(2)
    plt.plot(x, y2, label='w = -[1, 2, 3]')
    plt.title('plot 2')
    plt.legend(loc='upper left')
    plt.grid()
    plot2.show()

    input()
```

1.6) Consider a sample of 10 marbles drawn independently from a bin that holds red and green marbles. The probability of a red marble is μ. For $\mu = 0.05, = 0.5, and = 0.8$, compute the probability of getting no red marbles (v = 0) in the fol lowing cases.

a) We draw only one such sample. Compute the probability that v = 0.

b) We draw 1, 000 independent samples. Compute the probability that (at least) one of the samples has v = 0.

c) Repeat (b) for 1,000,000 independent samples.

Solution:

a) These experiments have a Binomial distribution, so:

$$X = \#\text{ red marbles in sample.} \quad X \sim B(10, \mu)$$

- For $\mu = 0.05$
$$P[X = 0] = 0.5987$$
- For $\mu = 0.5$
$$P[X = 0] = 9.8 \times 10^{-4}$$
- For $\mu = 0.8$
$$P[X = 0] = 1.02 \times 10^{-7}$$

b) These are new experiments which a Binomial distribution based on the previous experiment:

$$Y = \#\text{ samples with 0 red marbles} \quad Y \sim B(1000, \lambda)$$

- For $\lambda = 0.5987$
$$P[Y \geqslant 1] = 1$$
- For $\lambda = 9.8 \times 10^{-4}$
$$P[Y \geqslant 1] = 0.62487$$
- For $\lambda = 1.02 \times 10^{-7}$
$$P[Y \geqslant 1] = 1 \times 10^{-4}$$

c) Here is the same experiment, therefore is the almost the same distribution. We just have to increase the number of samples. For samples n = 1,000,000.

$$Y = \#\text{ samples with 0 red marbles} \quad Y \sim B(1000000, \lambda)$$

- For $\lambda = 0.5987$
$$P[Y \geqslant 1] = 1$$
- For $\lambda = 9.8 \times 10^{-4}$
$$P[Y \geqslant 1] = 1$$
- For $\lambda = 1.02 \times 10^{-7}$
$$P[Y \geqslant 1] = 0.09697$$

1.10) Assume that $X = \{x_0, x_1, ..., x_N, x_{N+1}, ..., x_{N+M}\}$ and $Y = \{-1, +1\}$ with an unknown target function $f : X \to Y$. The training data set$\mathcal{D} = \{(x_1, y_1), ..., (x_N, y_N)\}$. Define the off-training-set error of a hypothesis h with respect to f by

$$E_{\text{off}}(\text{h, f}) = \frac{1}{M} \sum_{m=1} M[h(x_{N+m}) \neq f(x_{N+m}))]$$

a) Say f (x) = +1 for all x and

$$h(x) = \begin{cases} +1 & \text{for } x = x_k \text{ and k is odd and } 1 \leqslant k \leqslant M + N \\ \text{-1} & \text{otherwhise} \end{cases}$$

What is E_{off}(h, f)?

b) We say that a target function f can 'generate' D in a noiseless setting if $y_n = f(x_n)$ for all $(x_n, y_n) \in D$. For a fixed D of size N, how many possible $f : X \to Y$ can generate D in a noiseless setting?

c) For a given hypothesis h and an integer k between 0 and M, how many of those f in (b) satisfy $E_{\text{off}}(\text{h, f}) = \frac{k}{M}$?

d) For a given hypothesis h, if all those f that generate D in a noiseless setting are equally likely in probability, what is the expected off training-set error $\mathsf{E}_{\mathsf{f}}[E_{\text{off}}(\text{h, f})]$?

e) A deterministic algorithm A is defined as a procedure that takes D as an input, and outputs a hypothesis h = A(D). Argue that for any two deterministic algorithms $A_1 and A_2$.

$$\mathsf{E}_{\mathsf{f}}[E_{\text{off}}(A_1(D, f)] = \mathsf{E}_{\mathsf{f}}[E_{\text{off}}(A_2(D, f)]$$

,

Solution:

a) If we rewrite the conditions, we can sum up and observe that the only case when $f(x_{N+i}) \neq h(x_{N+i}) \quad \forall 1 \leqslant i \leqslant M$ is when x_{N+i} is odd. So:

$$E_{\text{off}}(\text{h, f}) = \frac{1}{M}[\text{Total number of odd between 1 and M}]$$

There are two cases:

$$\text{M is even} \quad \to E_{\text{off}}(\text{h, f}) = \frac{1}{M}\frac{M}{2} = \frac{1}{2}$$

$$\text{M is odd} \quad \to E_{\text{off}}(\text{h, f}) = \frac{1}{M}(\frac{M-1}{2} + 1) = \frac{1}{2} - \frac{1}{2M} + \frac{1}{M} = \frac{1}{2} + \frac{1}{2M}$$

b) If $|D| = N$ there are 2^n possibles target functions that can generate D in a noiseless setting.

c) What we really want to know is:

$$\frac{1}{M}\sum_{m=1} M[h(x_{N+m}) \neq f(x_{N+m}))] = k$$

And this can sump up to the number of ways we have k errors in M trails. And this is a combination in the entire set:

$$C_k^M = \frac{M!}{k!(M-k)!}$$

d)

$$\mathsf{E}_\mathsf{f}[E_{\text{off}}(\text{h, f})] = \frac{1}{M}\mathsf{E}_\mathsf{f}[\text{Number of } h(x_{N+m}) \neq f(x_{N+m}))]$$

$$\frac{1}{M} \times M \times \frac{1}{2^M} = \frac{1}{2^M}$$

1.11) The matrix which tabulates the cost of various errors for the CIA and Supermarket applications in Example 1.1 is called a risk or loss matrix. For the two risk matrices in Example 1.1, explicitly write down the in sample error E_{in} that one should minimize to obtain g. This in-sample error should weight the different types of errors based on the risk matrix.

Solution:

a) Supermarket:

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

$$= \frac{1}{N}\sum_{y_n=1} e(h(x_n, 1)) + \sum_{y_n=-1} e(h(x_n, -1))$$

$$= \frac{1}{N}\sum_{y_n=1} 10[[h(x_n \neq 1)]] + \sum_{y_n=-1} [[h(x_n \neq -1)]]$$

b) CIA:

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

$$= \frac{1}{N}\sum_{y_n=1} e(h(x_n, 1)) + \sum_{y_n=-1} e(h(x_n, -1))$$

$$= \frac{1}{N}\sum_{y_n=1} [[h(x_n \neq 1)]] + \sum_{y_n=-1} 1000[[h(x_n \neq -1)]]$$