题记

今年2月的时候开始学习台大林轩田老师的机器学习课程，感觉讲的非常好，课程的参考教材是**learning from data**，网上查阅资料的时候发现关于这本书的笔记几乎没有，所以想自己做一个学习笔记，记录教材中的习题的解法，一来可以加深自己的理解，而来也可以给后来学习的小伙伴一些参考。 这份笔记主要以**learning from data**的习题解析为主，笔记形式为Markdown以及Jupyter Notebook结合的形式。因为笔者水平有限，难免有些错误，欢迎指出。

我的github地址：
https://github.com/Doraemonzzz

个人主页：
http://doraemonzzz.com/

参考资料:
https://blog.csdn.net/a1015553840/article/details/51085129
http://www.vynguyen.net/category/study/machine-learning/page/6/
http://book.caltech.edu/bookforum/index.php
http://beader.me/mlnotebook/

下面进入正题，这一次讨论第一章的讨论题和课后习题。

Chapter 1 The Learning Problem

Part 1:Exercise

Exercise 1.1 (Page 4)

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

(a) Medical diagnosis: A patient walks in with a medical history and some symptoms, and you want to identify the problem.
(b) Handwritten digit recognition (for example postal zip code recognition for mail sorting) .
(c) Determining if an email is spam or not.
(d) Predicting how an electric load varies with price, temperature, and day of the week.
(e) A problem of interest to you for which there is no analytic solution,but you have data from which to construct an empirical solution

这题考察了对learning from data框架的理解,框架可以参考课本第4页。

标识	输入空间	输出空间	目标函数
a	病史，症状	是否患病	判别函数(医疗资料至是否患病的映射)
b	手写数字	判别出来的数字	判别函数(手写数字至数字的映射)
c	邮件内容	是否为垃圾邮件	判别函数(邮件内容至是否为垃圾邮件的映射)
d	电力负载量，电力价格，温度，日期	电量的预测	预测函数
e	数据	处理结果	经验解决方案(empirical solution)

Exercise 1.2 (Page 6)

Suppose that we use a perceptron to detect spam messages. Let's say that each email message is represented by the frequency of occurrence of keywords, and the output is if the message is considered spam.

(a) Can you think of some keywords that will end up with a large positive weight in the perceptron?
(b) How about keywords that will get a negative weight?
(c) What parameter in the perceptron directly affects how many borderline messages end up being classified as spam?

这题考察了感知机的一些概念。

(a) 哪些关键字在感知机的模型中有较大权重？ 免费，打折，钱等

(b) 哪些关键字在感知机的模型中有负的权重？ 显然是和垃圾邮件无关的，例如工作，开会，作业等

(c) 感知机中哪些参数直接影响了边界信息最终被分类为垃圾邮件? 这里回顾下感知机模型：

$$h(x) = \text{sign}((\sum_{i=1}^{d} w_i x_i) + b)$$

所以这里直接影响的参数为 $w_i(i=1,2...d)$以及b

Exercise 1.3 (Page 8)

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

(a) Show that $y(t)w^T(t)x(t) < 0$. [Hint: $x(t)$ is misclassifed by $w(t)$.]
(b) Show that $y(t)w^T(t+1)x(t) > y(t)w^T(t)x(t)$. [Hint: Use (1.3).]
(c) As far as classifying $x(t)$ is concerned, argue that the move from $w(t)$ to $w(t+1)$ is a move 'in the right direction ' .

这里先对之前的感知机模型稍作变形，令$w_0 = b, x_0 = 1$，那么

$$h(x) = \text{sign}((\sum_{i=1}^{d} w_i x_i) + b)$$

可以变形为:

$$h(x) = \text{sign}((\sum_{i=0}^{d} w_i x_i)) = \text{sign}(w^T(t)x(t))$$

接着给出式(1.3)

对于$y(t) \neq \text{sign}(w^T(t)x(t))$的数据，更新规则是$w(t+1) = w(t) + y(t)x(t)$

(a) 因为$y(t) \neq \text{sign}(w^T(t)x(t))$，所以当$\text{sign}(w^T(t)x(t)) > 0$时，$y(t) = -1$，当$\text{sign}(w^T(t)x(t)) < 0$时，$y(t) = 1$，所以$y(t)w^T(t)x(t) < 0$

(b)

$$y(t)w^T(t+1)x(t) = y(t)(w(t) + y(t)x(t))x(t) = y(t)w^T(t)x(t) + y^2(t)x^T(t)x(t)$$

注意$x(t)$的第一个分量为1(参考课本)，所以$y^2(t)x^T(t)x(t) > 0$，因此

$$y(t)w^T(t+1)x(t) > y(t)w^T(t)x(t)$$

(c) 由(1)我们知道，如果分类错误，那么$y(t)w^T(t)x(t) < 0$，但利用(1.3)更新之后，$y(t)w^T(t+1)x(t) > y(t)w^T(t)x(t)$，也就是向着正方向前进了，所以如果资料是可分的，那么经过有限步之后可以得到w，使得对所有的x，$yw^Tx > 0$

Exercise 1.4 (Page 8)

Let us create our own target function f and data set $\mathcal{D}$ and see how the perceptron learning algorithm works. Take d = 2 so you can visualize the problem, and choose a random line in the plane as your target function,where one side of the line maps to 1 and the other maps to -1. Choose the inputs x_n of the data set as random points in the plane, and evaluate the target function on each x_n to get the corresponding output Y_n.

Now, generate a data set of size 20. Try the perceptron learning algorithm on your data set and see how long it takes to converge and how well the final hypothesis g matches your target f. You can find other ways to play with this experiment in Problem 1.4.

这题是对感知机模型编程实现一下，这里使用了Python，由于Problem 1.4和这题的代码差不多，这里把公共部分写在helper.py文件中，代码如下

```
# -*- coding: utf-8 -*-
"""
Created on Tue Feb 12 12:01:52 2019

@author: qinzhen
"""
```

```python
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei'] #用来正常显示中文标签
plt.rcParams['axes.unicode_minus'] = False #用来正常显示负号


def Judge(X, y, w):
    """
    判别函数，判断所有数据是否分类完成
    """
    n = X.shape[0]
    #判断是否同号
    num = np.sum(X.dot(w) * y > 0)
    return num == n

def data(N, d, rnd, t=0.1):
    """
    生成N个d维点（不包括偏置项1），x1+...+xd>=t的点标记为+1，x1+...+xd<=-t的点标记为-1，
    rnd为随机数生成器，形式为rnd = np.random.RandomState(seed)，seed为随机种子
    """
    X = []
    w = np.ones(d)
    while (len(X) < N):
        x = rnd.uniform(-1, 1, size=(d))
        if np.abs(x.dot(w)) >= t:
            X.append(x)

    X = np.array(X)
    y = 2 * (X.dot(w) > 0) - 1
    #添加第一个分量为1
    X = np.c_[np.ones((N, 1)), X]

    return X, y

def f(N, d, rnd, t=0.1, r=1):
    """
    生成N个d维点（不包括偏置项1），x1+...+xd>=t的点标记为+1，x1+...+xd<=-t的点标记为-1，
    rnd为随机数生成器，形式为rnd = np.random.RandomState(seed)，seed为随机种子
    利用PLA更新，如果r=1，那么按照顺序取点，否则随机取点
    """
    X, y = data(N, d, rnd, t=t)

    #记录次数
    s = 0
    #初始化w=[0, 0, 0]
    w = np.zeros(d + 1)
    #数据数量
    n = X.shape[0]
    if r == 1:
        while (Judge(X, y ,w) == 0):
            for i in range(n):
                if X[i, :].dot(w) * y[i] <= 0:
                    w += y[i] * X[i, :]
                    s += 1
```

```python
        else:
            while (Judge(X, y ,w) == 0):
                i = np.random.randint(0, N)
                if X[i, :].dot(w) * y[i] <= 0:
                    w += y[i] * X[i, :]
                    s += 1

    #直线方程为w0+w1*x1+w2*x2=0,根据此生成点
    a = np.arange(-1, 1, 0.1)
    b = (a * w[1] + w[0]) / (- w[2])

    #原直线方程为x1+x2 = 0
    c = - a

    #返回数据
    return a, b, c, X, y, s, w

def plot_helper(a, b, c, X, y, s, w, t=0):
    """
    作图函数
    """
    #画出图像
    plt.scatter(X[y == 1][:, 1], X[y == 1][:, 2], c='r', s=1)
    plt.scatter(X[y == -1][:, 1], X[y == -1][:, 2], c='b', s=1)
    plt.plot(a, b, label="("+str(w[0])+")+("+str(w[1])+")x1+("+str(w[2])+")x2=0")
    plt.plot(a, c, label="x1+x2="+str(t))
    plt.title(u"经过"+str(s)+u"次迭代收敛")
    plt.legend()
    plt.show()
```
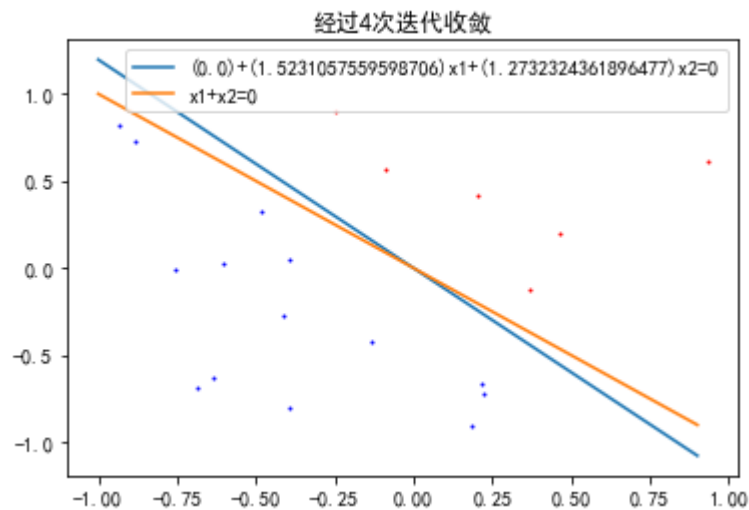
最后的代码如下

```python
# -*- coding: utf-8 -*-
"""
Created on Mon Feb 11 21:32:17 2019

@author: qinzhen
"""
import helper as hlp
import numpy as np
#%matplotlib inline

#设置随机种子，保证每次结果一致
seed = 42
rnd = np.random.RandomState(seed)
N = 20
d = 2
a, b, c, X, y, s, w = hlp.f(N, d, rnd)
hlp.plot_helper(a, b, c, X, y, s, w)
```

经过4次迭代收敛

$(0.0)+(1.5231057559598706) x1+(1.2732324361896477) x2=0$

$x1+x2=0$

Exercise 1.5 (Page 11)

Which of the following problems are more suited for the learning approach and which are more suited for the design approach?

(a) Determining the age at which a particular medical test should be performed
(b) Classifying numbers into primes and non-primes
(c) Detecting potential fraud in credit card charges
(d) Determining the time it would take a falling object to hit the ground
(e) Determining the optimal cycle for traffic lights in a busy intersection

learning approach是指机器学习的方法，design approach是指推导的方法，适合能够直接分析推导的问题

标识	适合的方式
a	learning approach
b	design approach
c	learning approach
d	design approach
e	learning approach

质数和自由落体问题可以直接得出结果，所以使用design approach

Exercise 1.6 (Page 14)

For each of the following tasks, identify which type of learning is involved (supervised, reinforcement, or unsupervised) and the training data to be used. If a task can fit more than one type, explain how and describe the training data for each type.

(a) Recommending a book to a user in an online bookstore
(b) Playing tic tac toe
(c) Categorizing movies into different types
(d) Learning to play music
(e) Credit limit: Deciding the maximum allowed debt for each bank customer

(a) 推荐系统的问题，非监督学习
(b) 游戏输赢是确定的，所以是监督学习，然后学习的过程是强化学习
(c) 标准的分类问题，监督学习
(d) 学习音乐的结果没有直接标志，非监督学习，然后学习的过程是强化学习
(e) 标准的预测问题，监督学习

Exercise 1.7 (Page 17)

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.

(a) $\mathcal{H}$ has only two hypotheses, one that always returns '●' and one that always returns 'o'. The learning algorithm picks the hypothesis that matches the data set the most.
(b) The same $\mathcal{H}$, but the learning algorithm now picks the hypothesis that matches the data set the least.
(c) $\mathcal{H} = \{\text{XOR}\}$ (only one hypothesis which is always picked), where XOR is defined by $\text{XOR}(x) = \bullet$ if the number of 1's in x is odd and $\text{XOR}(x) = o$ if the number is even.
(d) $\mathcal{H}$ contains all possible hypotheses (all Boolean functions on three variables), and the learning algorithm picks the hypothesis that agrees with all training examples, but otherwise disagrees the most with the XOR.

$\mathbf{x}$	y	g	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
0 0 0	o	o	o	o	o	o	o	o	o	o
0 0 1	●	●	●	●	●	●	●	●	●	●
0 1 0	●	●	●	●	●	●	●	●	●	●
0 1 1	o	o	o	o	o	o	o	o	o	o
1 0 0	●	●	●	●	●	●	●	●	●	●
1 0 1		?	o	o	o	o	●	●	●	●
1 1 0		?	o	o	●	●	o	o	●	●
1 1 1		?	o	●	o	●	o	●	o	●

先看下题目的背景，给定5组训练数据和结果，让我们预测其余3组数据的表现结果，现在假设在5组数据上全部正确，那么一共就有2^3种可能的函数，如上图所示。 现在题目分别给出4种假设空间和学习算法，这样可以得到结果g，我们需要给出和g在3个点均一致，2个点一致，1个点一致的的函数。

(a) 假设空间为全o以及全●，学习的算法是取和训练数据集符合程度最多的假设，显然全●在5组训练数据上有3组正确，全o只有2组正确，所以这里取全●

(b) 假设空间为全o以及全●，学习的算法是取和训练数据集符合程度最少的假设，所以这里取全o

(c) 这里的假设空间只有一个元素，有奇数个1则取●，否则取o

(d) 这里的假设空间是所有假设，学习算法是在训练数据上结果全部正确，但是在预测数据上和XOR不符合度最高的的假设，因此这里取和条件c结果完全相反的结果

x	条件a下的g	条件b下的g	条件c下的g	条件d下的g
101	●	o	o	●
110	●	o	o	●
111	●	o	●	o

所以题目的答案如下:

假设	3个点相同的函数	2个点相同的函数	1个点相同的函数
a	f_8	f_4, f_6, f_7	f_2, f_3, f_5
b	f_1	f_2, f_3, f_5	f_4, f_6, f_7
c	f_2	f_1, f_4, f_6	f_3, f_5, f_8
d	f_7	f_3, f_5, f_8	f_1, f_4, f_6

Exercise 1.8 (Page 19)

If $\mu = 0.9$, what is the probability that a sample of 10 marbles will have $\nu \leq 0.1$? [Hints: 1. Use binomial distribution. 2. The answer is a very small number.]

二项分布的题目，盒子例一共有两种球，90%是红球，10%是绿球，现在取10个，问红球数量小于等于1的概率。

$$P = 0.1^{10} + C_{10}^1 \times 0.1^9 \times 0.9 \approx 9 \times 10^{-9}$$

Exercise 1.9 (Page 19)

If $\mu = 0.9$, use the Hoeffding Inequality to bound the probability that a sample of 10 marbles will have $\nu \leq 0.1$ and compare the answer to the previous exercise.

回顾下Hoeffding不等式:

$$\mathbb{P}[|\mu - v| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

因此

$$
\begin{aligned}
\mathbb{P}[v \leq 0.1] &= \mathbb{P}[0.9 - v \geq 0.8] \\
&= \mathbb{P}[\mu - v \geq 0.8] \\
&\leq \mathbb{P}[|\mu - v| \geq 0.8] \\
&\leq 2e^{-2 \times 0.8^2 \times 10} \\
&\approx 5.5215451440744015 \times 10^{-6}
\end{aligned}
$$

这个比习题1.8的上界更松，因为Hoeffding不等式是通用的，所以上界肯定会更宽松一些

Exercise 1.10 (Page 23)

Here is an experiment that illustrates the difference between a single bin and multiple bins. Run a computer simulation for flipping 1000 fair coins. Flip each coin independently 10 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 three coins selected?

(b) Repeat this entire experiment a large number 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 $\mathbb{P}[|\mu - v| > \epsilon]$ as a function of ϵ, together with the Hoeffding bound $2e^{-2\epsilon^2 N}$ (on the same graph) .

(d) Which coins obey the Hoeffding bound, and which ones do not? Explain why.

(e) Relate part (d) to the multiple bins in Figure 1.10

介绍下这题的背景,先定义几个概念:

$$E_{in}(h) = (fraction\ of\ D\ where\ f\ and\ h\ disagree)$$

$$= \frac{1}{N} \sum_{i=1}^{n} [\![h(x_n) \neq f(x_n)]\!],$$

$$E_{out}(h) = \mathbb{P}[\![h(x) \neq f(x)]\!]$$

这里f是原假设,h是我们得到的假设，其次：

$$[\![statement]\!] = 1\ if\ the\ statement\ is\ true$$

$$[\![statement]\!] = 0\ if\ the\ statement\ is\ false$$

$E_{in}(h)$就是我们得到的假设和目标假设在训练集上的误差，$E_{out}(h)$是实际误差，所以$E_{out}(h)$相当于Hoeffding不等式中的μ，$E_{in}(h)$相当于Hoeffding不等式中的v。 因此：

$$\mathbb{P}[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

所以如果数据量够大，$E_{in}(h)$和$E_{out}(h)$的差距会越来越小。但之前的情况都是只有一个假设，如果只有一个假设，也就不需要学习了，所以这里考虑了一系列假设$\mathcal{H} = \{h_1, h_2, \ldots, h_M\}$，我们要从中挑选一个比较好的假设$g$。 这题希望解释的就是我们不能得出：

$$\mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

实际上我们只能得出：

$$\mathbb{P}[|E_{in}(g) - E_{out}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$$

更细节的部分可以参考书本。

回到原题，v_1, v_{rand}, v_{min}分别是第一个抛的硬币，随机抽取的硬币，抛十次之后得到正面数量最少的硬币（如果这样的硬币有多个，则取次序最前面的一个硬币）在十次投掷中正面的比例，μ是抛以上三种硬币得到正面的概率。

(a) 因为硬币是公平的，所以$\mu = 0.5$

(b) 首先根据题目要求模拟结果:

```python
# -*- coding: utf-8 -*-
"""
Created on Mon Feb 11 23:05:49 2019

@author: qinzhen
"""

import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] #用来正常显示中文标签
plt.rcParams['axes.unicode_minus']=False #用来正常显示负号

#首先模拟一次实验的结果，n表示硬币数量，1表示正面，0表示反面，最后返回的结果为正面的数量
def simu(n):
    #n个硬币，每个硬币投10次
    X = np.random.randint(0, 2, (n, 10))
    #计算每个硬币正面的数量
    X1 = np.sum(X, axis=1)
    #第一个元素
    y1 = X1[0]
    #随机元素
    y2 = X1[np.random.randint(0, n)]
    #最小值
    y3 = np.min(X1)
    return y1, y2, y3

#记录正面硬币得分的具体情况，存入一个长度为11的列表，第i个元素表示得分为i的次数
Y1 = [0]*11
Y2 = [0]*11
Y3 = [0]*11
#硬币数量
n = 1000
#实验次数
m = 10000
for i in range(m):
    y1, y2, y3 = simu(n)
    Y1[y1] += 1
    Y2[y2] += 1
    Y3[y3] += 1



#计算正面的次数
def total(x):
    s = 0
    for i in range(len(x)):
        s += i*x[i]
    return s
#投硬币的总次数
t = m * 10

####(b)
```
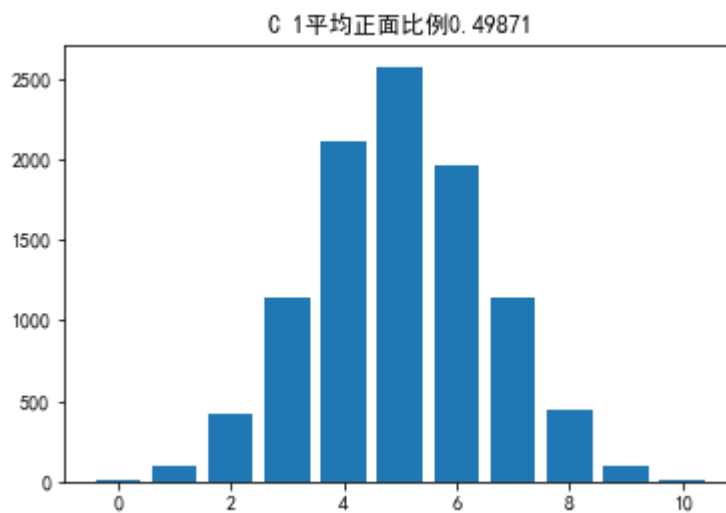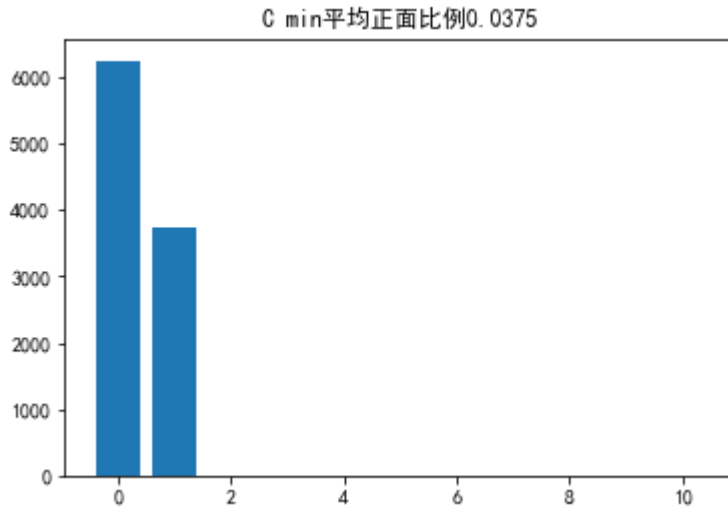
C 1平均正面比例0.49871

C rand平均正面比例0.49923

C min平均正面比例0.0375
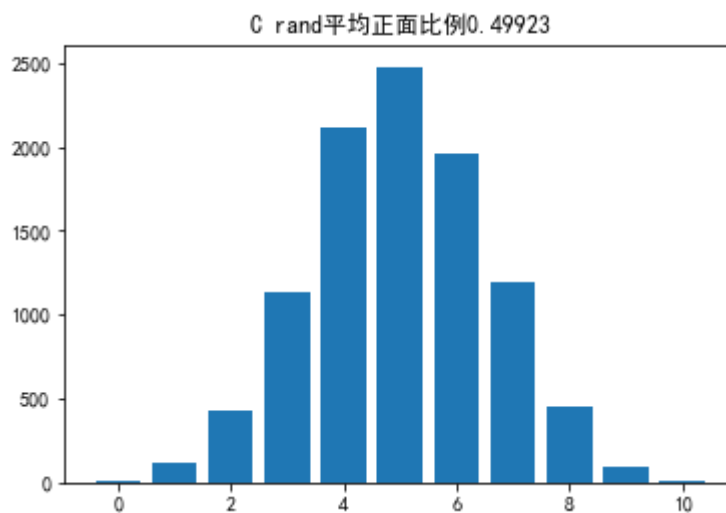
不出意外,$v_1, v_{\rm rand}$的平均得分率为0.5左右，而$v_{\rm min}$的平均得分率较低。事实上，我们可以求出$\mathbb{E}[v_{\rm min}]$,这里来分析一下：　这里有1000个得分率样本$X_1, X_2, \ldots, X_{1000}$，所以这里

$$\mathbb{E}[v_{\rm min}] = \mathbb{E}[\min(X_1, X_2, \ldots, X_{1000})]$$

下面计算数学期望,首先给出分布函数：

$$\mathbb{P}(v_{\rm min} \leq \frac{k}{10}) = 1 - \mathbb{P}(v_{\rm min} \geq \frac{k+1}{10})$$
$$= 1 - \mathbb{P}(\min(X_1, X_2, \ldots, X_{1000}) \geq \frac{k+1}{10})$$
$$= 1 - (\sum_{i=k+1}^{10} C_{10}^i (\frac{1}{2})^{10})^{1000}$$

所以：

$$\mathbb{P}(v_{\rm min} = \frac{k}{10}) = \mathbb{P}(v_{\rm min} \leq \frac{k}{10}) - \mathbb{P}(v_{\rm min} \leq \frac{k-1}{10})$$
$$= 1 - (\sum_{i=k+1}^{10} C_{10}^i (\frac{1}{2})^{10})^{1000} - [1 - (\sum_{i=k}^{10} C_{10}^i (\frac{1}{2})^{10})^{1000}]$$
$$= (\sum_{i=k}^{10} C_{10}^i (\frac{1}{2})^{10})^{1000} - (\sum_{i=k+1}^{10} C_{10}^i (\frac{1}{2})^{10})^{1000}$$

因此：

$$\mathbb{E}(v_{\rm min}) = \sum_{k=0}^{10} \frac{k}{10} \times P(v_{\rm min} = k)$$
$$= \sum_{k=0}^{10} \frac{k}{10} \times [(\sum_{i=k}^{10} C_{10}^i (\frac{1}{2})^{10})^{1000} - (\sum_{i=k+1}^{10} C_{10}^i (\frac{1}{2})^{10})^{1000}]$$

这里使用Python计算结果，最后得到$\mu \approx 0.037644419141365526$，可以看到和之前模拟的结果一致

```python
from scipy.special import comb

#P(v<=k),n表示一共几个硬币
def f(k, n):
    s = 0
    for i in range(k+1, 11):
        s += comb(10,i) / (2**10)
    return 1- s ** n

#P(v=k)
def g(k, n):
    return f(k, n) - f(k-1, n)

def h(n):
    s = 0
    for k in range(11):
        s += k / 10.0 * g(k, n)
    return s


print(h(1000))
```

```
0.03764441914136553
```

(c) 这里根据上述的结果作图

```python
epsilon = np.arange(0, 2, 0.01)
size = epsilon.shape[0]

Z1 = np.zeros(size)
Z2 = np.zeros(size)
Z3 = np.zeros(size)

#计算P(|u-v|>epsilon)
for i in range(size):
    for j in range(11):
        if abs((j - 5) / 10.0) > epsilon[i]:
            Z1[i] += Y1[j]
            Z2[i] += Y2[j]
            Z3[i] += Y3[j]
Z1 = Z1 / m
Z2 = Z2 / m
Z3 = Z3 / m

#Hoeffding上界值
Z = np.array([2 * np.exp(-2*(i**2)*10) for i in epsilon])

#作图
plt.plot(epsilon, Z, label=u"Hoeffding上界")
plt.plot(epsilon, Z1, label=u"P(|v1-u|)")
plt.plot(epsilon, Z2, label=u"P(|vrand-u|)")
plt.plot(epsilon, Z3, label=u"P(|vmin-u|)")
plt.xlabel(u'epsilon')
```
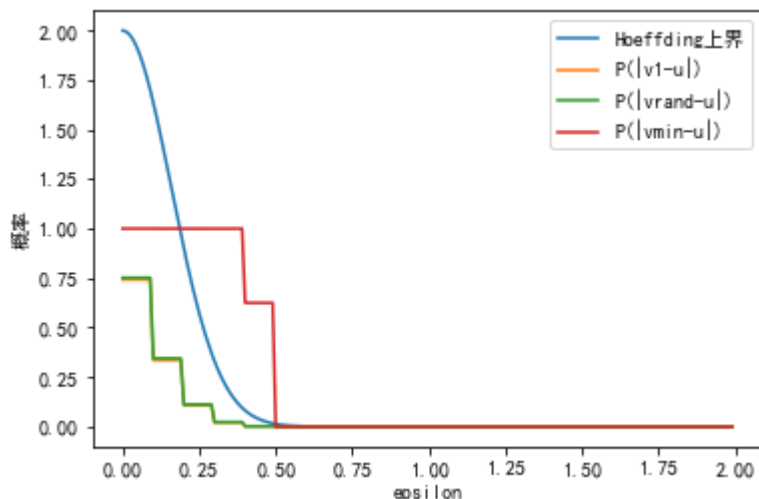
```
plt.ylabel(u'概率')
plt.legend()
plt.show()
```


(d) (e)首先由对称性不难发现v_1, v_{rand}实际上是等价的。从上图可以看到计算出来的$\mathbb{P}(|v_{\min} - \mu| > \epsilon)$比Hoeffding上界值大，所以$v_{\min}$违背了Hoeffding不等式，这是因为得到$v_{\min}$的过程如下

$$\text{选择 } X_i \text{如果 } X_i < X_j (\forall j \neq i)$$

其中$X_1, \ldots, X_{1000}$是每个硬币正面的比例。所以上述过程相当于1000个假设，因此不能使用不等式

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

应该使用不等式

$$\mathbb{P}[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2Me^{-2\epsilon^2 N}$$

其中$M = 1000$。

Exercise 1.11 (Page 25)

We are given a data set $\mathcal{D}$ of 25 training examples from an unknown target function $f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X} = \mathbb{R}$ and $\mathcal{Y}$= {-1, +1}. To learn f, we use a simple hypothesis set $\mathcal{H} = \{h_1, h_2\}$ where h_1 is the constant +1 function and h_2 is the constant -1.

We consider two learning algorithms, S (smart) and C (crazy). S chooses the hypothesis that agrees the most with $\mathcal{D}$ and C chooses the other hypothesis deliberately. Let us see how these algorithms perfrm out of sample from the deterministic and probabilistic points of view. Assume in the probabilistic view that there is a probability distribution on $\mathcal{X}$, and let $\mathbb{P}[f(x) = +1] = p$.

(a) Can S produce a hypothesis that is guaranteed to perform better than random on any point outside D?
(b) Assume for the rest of the exercise that all the examples in $\mathcal{D}$ have $y_n = +1$. Is it possible that the C hypothesis that produces turns out to be better than the hypothesis that S produces?
(c) If $p = 0.9$, what is the probability that S will produce a better hypothesis than C?
(d) Is there any value of p for which it is more likely than not that C will produce a better hypothesis than S?

(a) 利用S算法会选择在训练集上效果较好的假设，但是并不能保证在训练集以外的部分表现的很好

(b) 这题及之后的习题假设在训练集上的$y_n = +1$均为1，利用C算法得到的假设在训练集上的准确率为0，利用S算法得到的假设在训练集上的准确率为1，但是在训练集以外的部分未知，所以C算法得到的假设仍然有可能比S算法得到的假设表现好

(c) 令S产生的结果为f_s,C产生的结果为f_c，需要求得的概率为

$$\mathbb{P}(\mathbb{P}[f_s = f] > \mathbb{P}[f_c = f])$$

因为$f_s(x)$恒等于1，$\mathbb{P}[f(x) = +1] = 0.9$，所以

$$\mathbb{P}[f_s = f] = \mathbb{P}[f(x) = +1] = p = 0.9$$
$$\mathbb{P}[f_c = f] = 1 - \mathbb{P}[f_s = f] = 0.1$$

因此

$$\mathbb{P}(\mathbb{P}[f_s = f] > \mathbb{P}[f_c = f]) = \mathbb{P}(0.9 > 0.1) = 1$$

(d) 这题是上题的一般化结果,题目实际上问的是是否存在p，使得

$$\mathbb{P}(\mathbb{P}[f_s = f] > \mathbb{P}[f_c = f]) < 0.5$$

因为

$$\mathbb{P}[f(x) = +1] = p$$

所以

$$\mathbb{P}[f_s = f] = p, \mathbb{P}[f_c = f] = 1 - p$$
$$\mathbb{P}(\mathbb{P}[f_s = f] > \mathbb{P}[f_c = f]) = \mathbb{P}(p > 1 - p)$$

所以当$p < 0.5$时，

$$\mathbb{P}(\mathbb{P}[f_s = f] > \mathbb{P}[f_c = f]) = 0 < 0.5$$

因此本题结果为$p < 0.5$

Exercise 1.12 (Page 26)

A friend comes to you with a learning problem. She says the target function is completely unknown, but she has 4, 000 data points. She is willing to pay you to solve her problem and produce for her a g which approximates f. What is the best that you can promise her among the following:

(a) After learning you will provide her with a g that you will guarantee approximates f well out of sample.
(b) After learning you will provide her with a g,and with high probability the g which you produce will approximate f well out of sample.
(c) One of two things will happen.
  (i) You will produce a hypothesis g;
  (ii) You will declare that you failed .
  If you do return a hypothesis g, then with high probability the g which you produce will approximate well out of sample.

这里选(c),因为不能保证学习出一个比较好的假设

Exercise 1.13 (Page 31)

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

$$P(y|x) = \begin{cases} \lambda & y = f(x) \\ 1 - \lambda & y \neq f(x) \end{cases}$$

(a) What is the probability of error that h makes in approximating y?
(b) At what value of λ will the perfrmance of h be independent of μ?
   [Hint: The noisy target will look completely random.]

(a)错误有两种，一种是"去真":$\lambda\mu$，另一种是"取伪"：$(1-\lambda)(1-\mu)$，所以犯错的概率一共$\lambda\mu + (1-\lambda)(1-\mu)$

(b)

$$P = \lambda\mu + (1-\lambda)(1-\mu) = \lambda\mu + 1 - \mu - \lambda + \lambda\mu = 1 - \lambda + (2\lambda - 1)\mu$$

所以当$\lambda = \frac{1}{2}$，$P = 1 - \lambda$，因此和μ独立

Part 2:Problems

Problem 1.1 (Page 33)

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? Hint: Use Bayes' Theorem:$\mathbb{P}[A \text{ and } B] = \mathbb{P}[A|B]\mathbb{P}[B] = \mathbb{P}[B|A]\mathbb{P}[A]$

这题还是比较基础的，有两个包，一个包里有两个黑球，另一个包里有一个黑球和一个白球，先随机取一个包，再随机从这个包里随机取一个球，当发现第一取出的球是黑球时，再从这个包里再取一次球，现在问第二次取出的球也是黑色的概率? 利用贝叶斯公式即可，记第一次拿到黑球的事件为A，连续两次拿到黑球的事件为B，所以概率为

$$\begin{aligned} p &= \mathbb{P}(B|A) \\ &= \frac{\mathbb{P}(AB)}{\mathbb{P}(A)} \\ &= \frac{\mathbb{P}(B)}{\mathbb{P}(A)} \\ &= \frac{\frac{1}{2} \times \frac{1}{2}}{\frac{1}{2} + \frac{1}{2} \times \frac{1}{2}} \\ &= \frac{2}{3} \end{aligned}$$

Problem 1.2 (Page 33)

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

(a) 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 $x_2 = ax_1 + b$, what are the slope a and intercept b in terms of w_0, w_1, w_2?

(b) Draw a picture for the cases $w = [1, 2, 3]^T$ and $w = -[1, 2, 3]^T$.

In more than two dimensions, the +1 and - 1 regions are separated by a hyperplane, the generalization of a line.

(a) 由解析几何知识，$w^T x = 0$直线的一侧每个点$w^T x > 0$，另一侧的每个点$w^T x < 0$，因此$h(x) = +1$和 $h(x) = -1$被一条直线所分开。

直线方程为

$$w^T x = w_0 + w_1 x_1 + w_2 x_2 = 0$$

所以当$w_2 \neq 0$时
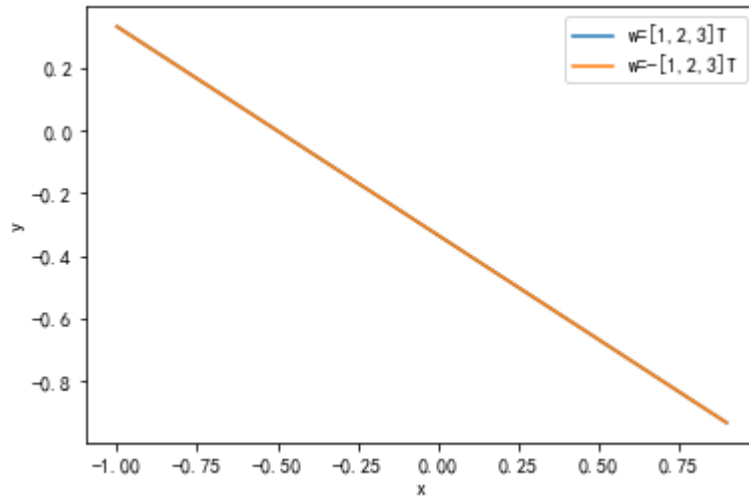
$$x_2 = -\frac{w_0}{w_2} - \frac{w_1}{w_2} x_1$$

(b)用Python作图

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Feb 12 10:43:54 2019

@author: qinzhen
"""

import matplotlib.pyplot as plt
import numpy as np

x = np.arange(-1, 1, 0.1)
y1 = np.array([-1/3 - 2/3 * i for i in x])
y2 = np.array([-1/3 - 2/3 * i for i in x])
plt.plot(x, y1, label='w=[1,2,3]T')
plt.plot(x, y2, label='w=-[1,2,3]T')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```

可以看到两条直线重合，这是由$x_2 = -\frac{w_0}{w_2} - \frac{w_1}{w_2}x_1$所决定的。

Problem 1.3 (Page 33)

Prove that the PLA eventually converges to a linear separator for separable data. The following steps will guide you through the proof. Let w^* be an optimal set of weights (one which separates the data).The essential idea in this proof is to show that the PLA weights $w(t)$ get "more aligned" with w^* with every iteration . For simplicity, assume that $w(0) = 0$.

(a) Let $\rho = \min_{1 \leq n \leq N} y_n\left(w^{*T}x_n\right)$. Show that $\rho > 0$.

(b) Show that $w^T(t)w^* \geq w^T(t-1)w^* + \rho$, and conclude that $w^T(t)w^* \geq t\rho$.[Hint: Use induction.]

(c) Show that $\|w(t)\|^2 \leq \|w(t-1)\|^2 + \|x(t-1)\|^2$.
[Hint: $y(t-1) \cdot \left(w^T(t-l)x(t-1)\right) \leq 0$ because $x(t-1)$ was misclassified by $w(t-1)$].

(d) Show by induction that $\|w(t)\|^2 \leq tR^2$, where$R = \max_{1 \leq n \leq N}\|xn\|$ ·

(e) Using (b) and (d), show that

$$\frac{w(t)}{\|w(t)\|}w^* \geq \sqrt{t}\frac{\rho}{R}$$

and hence prove that

$$t \leq \frac{R^2\|w^*\|^2}{\rho^2}$$

[Hint: $\frac{w(t)w^*}{\|w(t)\| \; \|w^*\|} \leq 1$. Why?]

In practice, PLA converges more quickly than the bound $\frac{R^2\|w^*\|^2}{\rho^2}$ suggests.Nevertheless, because we do not know ρ in advance, we can 't determine the number of iterations to convergence, which does pose a problem if the data is non-separable.

(a) 因为可分,所以$\forall\, n(1 \leq n \leq N)$，$y_n\left(w^{*T}x_n\right) > 0$，然后因为$N$有限，所以$\rho = \min_{1 \leq n \leq N}y_n\left(w^{*T}x_n\right) > 0$

(b)先回顾下更新规则：

$$\text{对于}y(t) \neq \text{sign}(w^T(t)x(t))\text{的数据,更新规则是 } w(t+1) = w(t) + y(t)x(t)$$

因此

$$
\begin{aligned}
w^T(t)w^* &= (w^T(t-1) + y(t-1)x^T(t-1))w^* \\
&= w^T(t-1)w^* + y(t-1)x^T(t-1)w^* \\
&= w^T(t-1)w^* + y(t-1)x(t-1)w^{*T} \\
&\geq w^T(t-1)w^* + \rho
\end{aligned}
$$

这里$y(t-1)x^T(t-1)w^* = y(t-1)x(t-1)w^{*T}$是因为$x^T(t-1)w^*$是一个实数，所以它的转置$x(t-1)w^{*T}$等于它本身，最后一步是因为

$$y(t-1)x(t-1)w^{*T} \geq \min_{1 \leq n \leq N} y_n(w^{*T}x_n) = \rho$$

再来证明 $w^T(t)w^* \geq t\rho$，利用数学归纳法 $t = 0$时,$w^T(t)w^* = w^T(0)w^* = 0, t\rho = 0$，因此$t = 0$时结论成立
假设$t = k$时结论成立，$t = k+1$时，由之前证明的结论知

$$w^T(k+1)w^* \geq w^T(k)w^* + \rho \geq k\rho + \rho = (k+1)\rho$$

所以$t = k+1$时结论成立

(c)对$w(t+1) = w(t) + y(t)x(t)$两边取模的平方，注意这里$||x||^2 = x^T x, x \in \mathbb{R}^n$

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

利用实数的转置等于其本身以及$y(t) = \pm 1$，可得

$$
\begin{aligned}
y(t)x^T(t)w(t) &= w^T(t)y(t)x(t) \\
y(t)x^T(t)y(t)x(t) &= y^2(t)x^T(t)x(t) = x^T(t)x(t)
\end{aligned}
$$

所以

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

因为误分，所以

$$y(t)x^T(t)w(t) \leq 0$$

因此

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

(d)利用数学归纳法，注意$w(0) = 0$
当$t = 1$时，由(e)知

$$||w(1)||^2 \leq ||w(0)||^2 + ||x(0)||^2 \leq R^2$$

假设$t = k$时结论成立，当$t = k + 1$时：

$$||w(k+1)||^2 \leq ||w(k)||^2 + ||x(t)||^2 \leq kR^2 + R^2 = (k+1)R^2$$

因此$t = k + 1$时结论成立

(e)由(d)知$||w(t)||^2 \leq tR^2$，所以

$$||w(t)|| \leq \sqrt{t}R$$

由(b)知$w^T(t)w^* \geq t\rho$，所以

$$\frac{w^T(t)w^*}{||w(t)||} \geq \frac{t\rho}{\sqrt{t}R} = \sqrt{t}\frac{\rho}{R}$$

由Cauthy-Schwarz不等式知

$$w^T(t)w^* \leq ||w(t)||\,||w^*||$$

因此

$$\sqrt{t}\frac{\rho}{R} \leq ||w^*||$$
$$t \leq \frac{||w^*||^2 R^2}{\rho^2}$$

这意味着t是一个有限的数，所以PLA一定会收敛

Problem 1.4 (Page 34)

In Exercise 1.4, we use an artificial data set to study the perceptron learning algorithm . This problem leads you to explore the algorithm further with data sets of different sizes and dimensions.

(a) Generate a linearly separable data set of size 20 as indicated in Exercise 1.4. Plot the examples $\{(x_n, y_n)\}$ as well as the target function f on a plane. Be sure to mark the examples from different classes differently,and add labels to the axes of the plot.
(b) Run the perceptron learning algorithm on the data set above. Report the number of updates that the algorithm takes before converging. Plot the examples $\{(x_n, y_n)\}$, the target function f, and the final hypothesis g in the same figure. Comment on whether f is close to g.
(c) Repeat everything in (b) with another randomly generated data set of size 20. Compare your results with (b) .
(d) Repeat everything in (b) with another randomly generated data set of size 100. Compare your results with (b) .
(e) Repeat everything in (b) with another randomly generated data set of size 1, 000. Compare your results with (b) .
(f) Modify the algorithm such that it takes $x_n \in \mathbb{R}^{10}$ instead of $\mathbb{R}^2$. Randomly generate a linearly separable data set of size 1, 000 with $x_n \in \mathbb{R}^{10}$ and feed the data set to the algorithm. How many updates does the algorithm take to converge?
(g) Repeat the algorithm on the same data set as (f) for 100 experiments. In the iterations of each experiment, pick $x(t)$ randomly instead of deterministically. Plot a histogram for the number of updates that the algorithm

takes to converge.

(h) Summarize your conclusions with respect to accuracy and running time as a function of N and d.

(a)(b)(c)这三题和Exercise 1.4基本一致，稍微不同的是需要画出目标函数f以及更新次数

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Feb 12 11:41:30 2019

@author: qinzhen
"""
import helper as hlp
import numpy as np
import matplotlib.pyplot as plt

#设置随机种子，保证每次结果一致
seed = 42
rnd = np.random.RandomState(seed)

#(a)(b)(c)
N = 20
d = 2
a, b, c, X, y, s, w = hlp.f(N, d, rnd)
hlp.plot_helper(a, b, c, X, y, s, w)
```
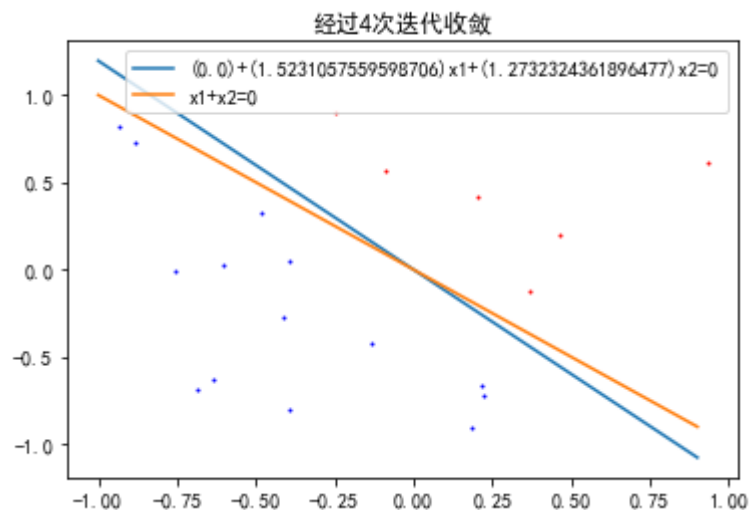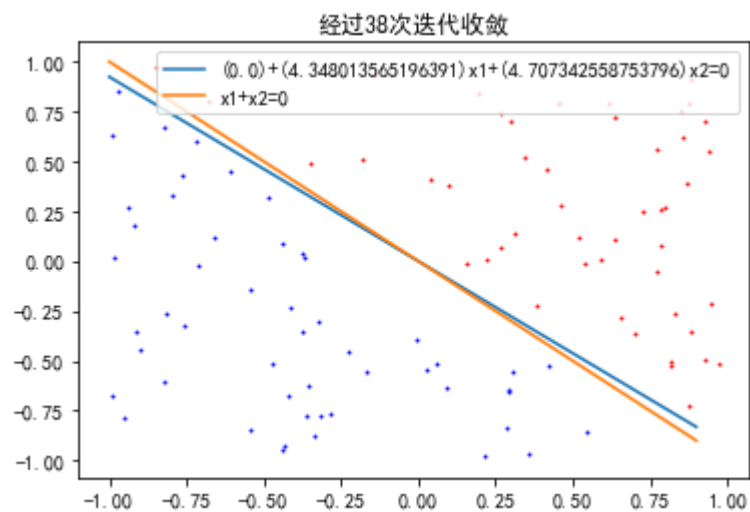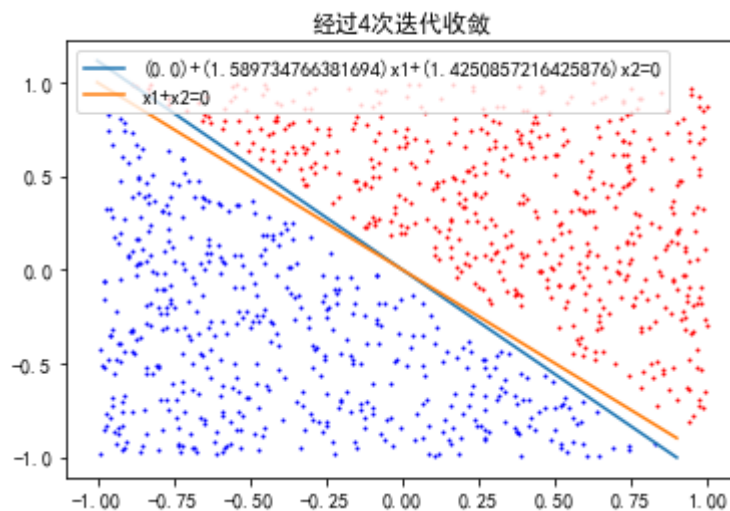

(d)数据量改为100

```python
#(d)
N = 100
a, b, c, X, y, s, w = hlp.f(N, d, rnd)
hlp.plot_helper(a, b, c, X, y, s, w)
```

(e)数据量修改为1000

```
#(e)
N = 1000
a, b, c, X, y, s, w = hlp.f(N, d, rnd)
hlp.plot_helper(a, b, c, X, y, s, w)
```
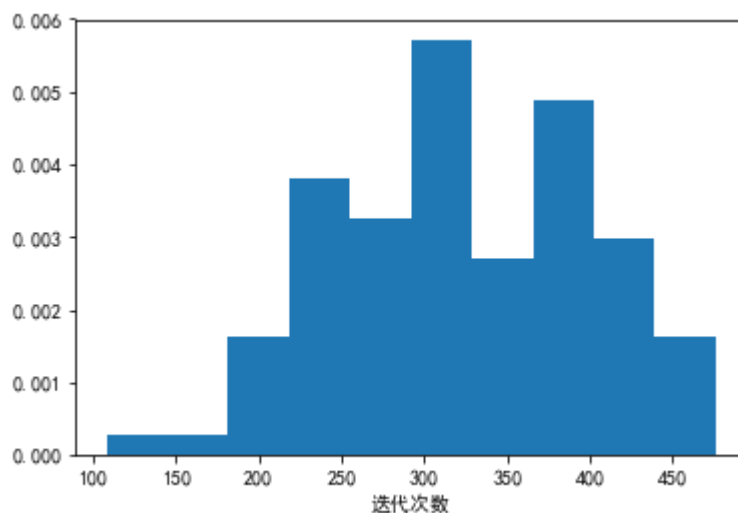

(f)这里将数据修改为$\mathbb{R}^{10}$

```
#(f)
#修改数据维度
N = 1000
d = 10
a, b, c, X, y, s, w = hlp.f(N, d, rnd)
print("迭代次数为" + str(s))
```

迭代次数为336

(g)重复(f)100次，做出直方图

```
#(g)
res = []
for i in range(100):
    a, b, c, X, y, s, w = hlp.f(N, d, rnd, r=0)
    res.append(s)
plt.hist(res, normed=True)
plt.xlabel("迭代次数")
plt.show()
```


(h)总结下来，随N和d的增加，迭代次数会增加。

Problem 1.5 (Page 35)

The perceptron learning algorithm works like this: In each iteration t, pick a random $(x(t), y(t))$ and compute the 'signal ' $s(t) = w^T(t)x(t)$.If $y(t) \cdot s(t) \leq 0$, update w by

$$w(t + 1) = w(t) + y(t) \cdot x(t)$$

One may argue that this algorithm does not take the 'closeness' between $s(t)$ and $y(t)$ into consideration. Let's look at another perceptron learning algorithm: In each iteration, pick a random $(x(t), y(t))$ and compute $s(t)$. If $y(t) \cdot s(t) \leq 1$, update w by

$$w(t + 1) = w(t) + \eta(y(t) - s(t)).x(t)$$

where η is a constant. That is, if $s(t)$ agrees with $y(t)$ well (their product is > 1), the algorithm does nothing. On the other hand, if $s(t)$ is further from $y(t)$, the algorithm changes $w(t)$ more. In this problem , you are asked to implement this algorithm and study its perfrmance.

(a) Generate a training data set of size 100 similar to that used in Exercise 1.4.Generate a test data set of size 10, 000 from the same process. To get g,run the algorithm above with $\eta = 100$ on the training data set, until a maximum of 1,000 updates has been reached. Plot the training data set, the target function f, and the final hypothesis g on the same figure. Report the error on the test set.
(b) Use the data set in (a) and redo everything with $\eta = 1$.
(c) Use the data set in (a) and redo everything with $\eta = 0.01$.
(d) Use the data set in (a) and redo everything with $\eta = 0.0001$.

(e) Compare the results that you get from (a) to (d) .

The algorithm above is a variant of the so called Adaline (Adaptive Linear Neuron) algorithm for perceptron learning.

(a)(b)(c)(d)这题的意思就是在更新的时候考虑"偏离距离"，并增加一个参数η，这里取二维平面上的点，下面通过编程实现一下。尝试的过程中发现η取100会导致数太大，所以这里取1,0.1,0.01,0.001

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Feb 12 13:02:55 2019

@author: qinzhen
"""
import numpy as np
import matplotlib.pyplot as plt
import helper as hlp
plt.rcParams['font.sans-serif']=['SimHei'] #用来正常显示中文标签
plt.rcParams['axes.unicode_minus']=False #用来正常显示负号

def Adaline(n1, n2, d, rnd, eta, w0, iteration=1000):
    """
    生成n1+n2个d维点（不包括偏置项1），n1个点作为训练数据，n2个点作为测试数据，
    x1+...+xd>=t的点标记为+1，x1+...+xd<=-t的点标记为-1，
    w0为分界线的法向量，w0=[0] + [1] * d，这里d = 2
    """
    #生成数据
    X_train, y_train = hlp.data(n1, d, rnd)
    X_test, y_test = hlp.data(n2, d, rnd)

    #记录次数
    T = 0
    w = np.zeros(d + 1)
    #print(X_train.dot(w) * y_train)
    while(hlp.Judge(X_train, y_train, w)==False and T < iteration):
        i = np.random.randint(0, n1)
        s = X_train[i, :].dot(w)
        a = s * y_train[i]
        if a <= 1:
            w += eta * (y_train[i] - s) * X_train[i, :]
            T += 1

    #计算错误率
    num = np.sum(X_test.dot(w) * y_test <= 0)

    print("n为"+str(eta)+"时错误率为"+str(num / n2))

    #直线方程为w0+w1*x+w2*y=0,根据此生成点
    X3 = np.arange(-1, 1, 0.1)
    Y3 = np.array([(X3[i]*w[1]+w[0])/(-w[2]) for i in range(len(X3))])

    #目标函数
    X4 = np.arange(-1, 1, 0.1)
    Y4 = np.array([(X3[i]*w0[1]+w0[0])/(-w0[2]) for i in range(len(X4))])
```
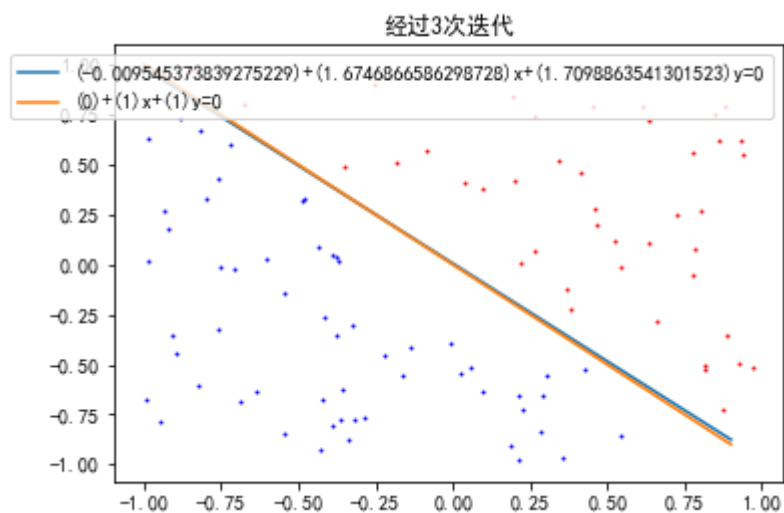
```
#画出图片
plt.scatter(X_train[y_train == 1][:, 1], X_train[y_train == 1][:, 2], c='r', s=1)
plt.scatter(X_train[y_train == -1][:, 1], X_train[y_train == -1][:, 2], c='b', s=1)
plt.plot(X3, Y3, label="("+str(w[0])+")+("+str(w[1])+")x+("+str(w[2])+")y=0")
plt.plot(X4, Y4, label="("+str(w0[0])+")+("+str(w0[1])+")x+("+str(w0[2])+")y=0")
plt.title(u"经过"+str(T)+u"次迭代")
#设置坐标范围
#plt.xticks(np.arange(0,10))
#plt.yticks(np.arange(0,10))
plt.legend()
plt.show()

#设置随机种子，保证每次结果一致
seed = 42
rnd = np.random.RandomState(seed)
Eta = [1, 0.1, 0.01, 0.001]
for eta in Eta:
    Adaline(100, 10000, 2, rnd, eta, [0, 1, 1])
```
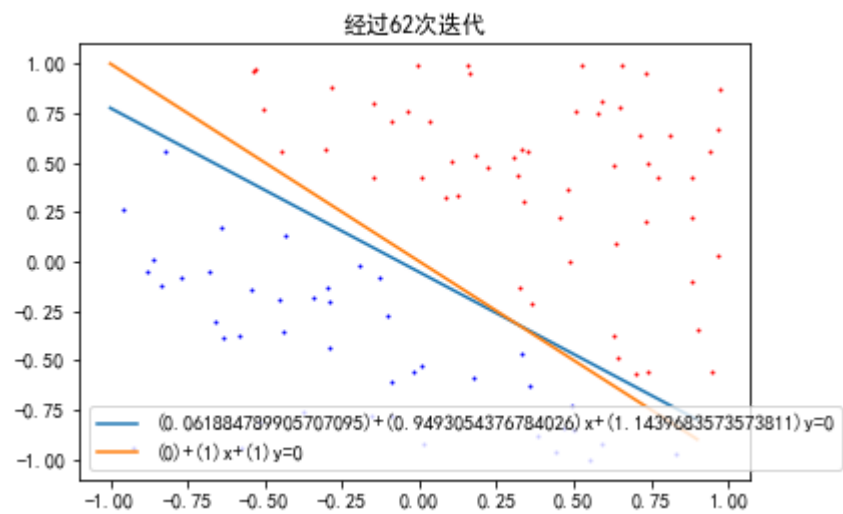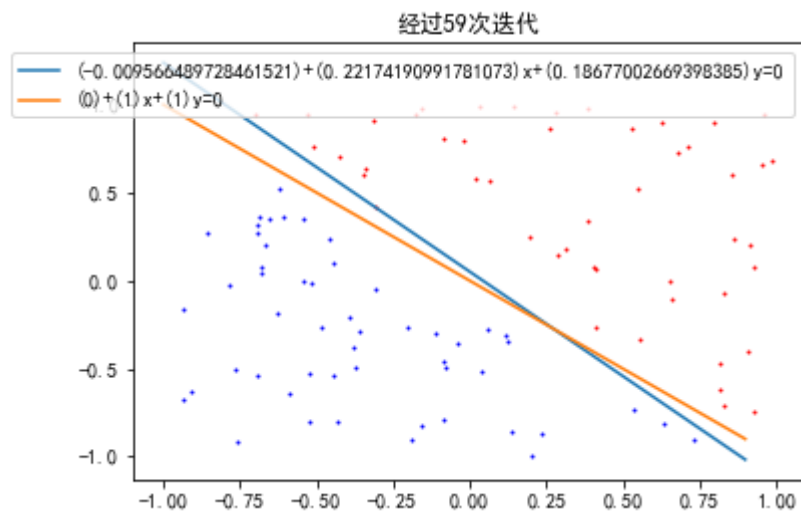
n为1时错误率为0.003

经过3次迭代

n为0.1时错误率为0.0153
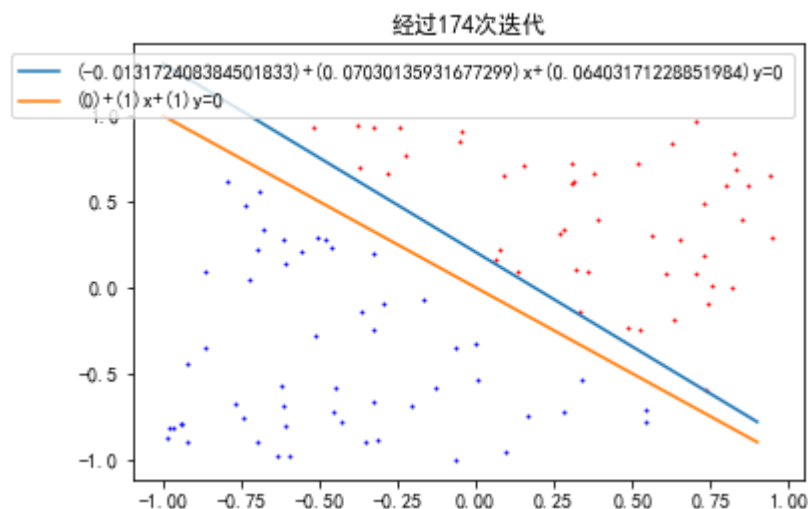
经过62次迭代

$(0.061884789905707095)+(0.9493054376784026)x+(1.1439683573573811)y=0$

$(0)+(1)x+(1)y=0$

n为0.01时错误率为0.0081

经过59次迭代

$(-0.009566489728461521)+(0.22174190991781073)x+(0.18677002669398385)y=0$

$(0)+(1)x+(1)y=0$

n为0.001时错误率为0.009

经过174次迭代

(f)η为每次更新的步长，从上述实验中可以看出，随着η减小，错误率先减小再增大，大约在$\eta = 0.01$处效果最好。

Problem 1.6 (Page 35)

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 $\mu = 0.05, \mu = 0.5$, and $\mu = 0.8$, compute the probability of getting no red marbles $(v = 0)$ in the following 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.

(a)抽出10个样本，$v = 0$的概率为$P_0 = (1 - \mu)^{10}$

(b)重复(a)1000次，至少有一次出现$v = 0$的概率为$P_1 = 1 - (1 - P_0)^{1000}$

(c)重复(a)1000,000次，至少有一次出现$v = 0$的概率为$P_2 = 1 - (1 - P_0)^{1000000}$

Problem 1.7 (Page 36)

A sample of heads and tails is created by tossing a coin a number of times independently. Assume we have a number of coins that generate different samples independently. For a given coin , let the probability of heads (probability of error) be μ. The probability of obtaining k heads in N tosses of this coin is given by the binomial distribution:

$$\mathbb{P}[k|N, \mu] = \binom{N}{k} \mu^k (1 - \mu)^{N-k}$$

Remember that the training error v is $\frac{k}{N}$.

(a) Assume the sample size (N) is 10. If all the coins have $\mu = 0.05$. Compute the probability that at least one coin will have $v = 0$ for the case of 1 coin, 1,000 coins, 1,000,000 coins. Repeat for $\mu = 0.8$.
(b) For the case $N = 6$ and 2 coins with $\mu = 0.5$ for both coins, plot the probability

$$\mathbb{P}[\max_i |v_i - \mu_i| > \epsilon]$$

for ϵ in the range $[0, 1]$ (the max is over coins) . On the same plot show the bound that would be obtained using the Hoeffding\ Inequality. Remember that for a single coin, the Hoeffding bound is

$$\mathbb{P}[|\mu - v| > \epsilon] \leq 2e^{-2\epsilon^2 N}$$

[Hint: Use $\mathbb{P}[A \ or \ B] = \mathbb{P}[A] + \mathbb{P}[B] - \mathbb{P}[A \ and \ B] = \mathbb{P}[A] + \mathbb{P}[B] - \mathbb{P}[A]\mathbb{P}[B]$, where the last equality follows by independence, to evaluate P[max . . .]]

(a)这题和上一题应该是差不多的意思，和上一题的(c)稍有不同。这里P_i表示i个硬币至少出现一次$v = 0$的概率

$$P_1 = (1 - \mu)^{10}$$
$$P_{1000} = 1 - (1 - P_1)^{1000}$$
$$P_{1000000} = 1 - (1 - P_1)^{1000000}$$

(b)注意$v = \frac{k}{N}$，$N = 6$且有两枚硬币。因此

$$\begin{aligned}
\mathbb{P}[\max_i |v_i - \mu_i| > \epsilon] &= 1 - \mathbb{P}[\max_i |v_i - \mu_i| \leq \epsilon] \\
&= 1 - (\mathbb{P}[|v - \mu| \leq \epsilon])^2 \\
&= 1 - (\mathbb{P}[|\frac{k}{N} - 0.5| \leq \epsilon])^2 \\
&= 1 - (\mathbb{P}[|k - 3| \leq 6\epsilon])^2 \\
&= 1 - (\mathbb{P}[3 - 6\epsilon \leq k \leq 3 + 6\epsilon])^2 \\
&= 1 - (\sum_{3-6\epsilon \leq k \leq 3+6\epsilon} \binom{N}{k} \mu^k (1 - \mu)^{N-k})^2
\end{aligned}$$

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Feb 12 11:03:25 2019

@author: qinzhen
"""

import numpy as np
import matplotlib.pyplot as plt
from scipy.special import comb
plt.rcParams['font.sans-serif']=['SimHei'] #用来正常显示中文标签
plt.rcParams['axes.unicode_minus']=False #用来正常显示负号

def Pmax(N, x, u):
    low = int(3 - 6*x)
    up = int(3 + 6*x)
    s = 0
    for k in range(low, up+1):
        s += comb(N, k) * (u ** k) * ((1 - u) ** (N - k))
    return 1 - s ** 2

x = np.arange(0, 1, 0.01)
y1 = np.array([Pmax(6, i, 0.5) for i in x])
y2 = np.array([2 * np.exp(-2 * 6 * (i ** 2)) for i in x])
```
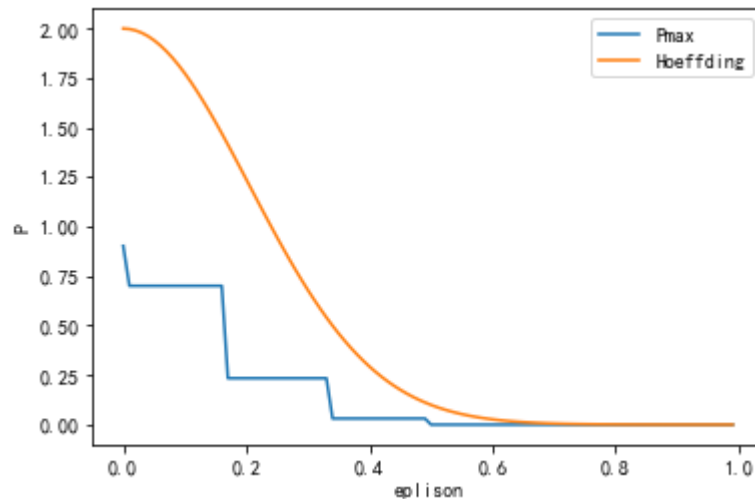
```
plt.plot(x, y1, label="Pmax")
plt.plot(x, y2, label="Hoeffding")
plt.xlabel("eplison")
plt.ylabel("P")
plt.legend()
plt.show()
```


Problem 1.8 (Page 36)

The Hoeffding Inequality is one form of the law\ of\ large\ numbers. One of the simplest forms of that law is the Chebyshev Inequality,which you will prove here.

(a) If t is a nonnegative random variable, prove that for any $\alpha > 0$, $\mathbb{P}[t \geq \alpha] \geq \frac{\mathbb{E}(t)}{\alpha}$

(b) If u is any random variable with mean μ and variance σ^2 , prove that for any $\alpha > 0$, $\mathbb{P}[(u - \mu)^2 \geq \alpha] \leq \frac{\sigma^2}{\alpha}$

[Hint: Use (a)]

(c) If $u_1, \ldots, u_N$ are iid random variables, each with mean μ and variance σ^2 ,and $u = \frac{1}{N} \sum_{n=1}^{N} u_n$, prove that for any $\alpha > 0$,

$$\mathbb{P}[(u - \mu)^2 \geq \alpha] \leq \frac{\sigma^2}{N\alpha}$$

Notice that the RHS of this Chebyshev Inequality goes down linearly in N,while the counterpart in Hoeffding's Inequality goes down exponentially. In Problem 1 .9, we develop an exponential bound using a similar approach.

这题是有关切比雪夫不等式的，下面分别证明下

(a)

$$\mathbb{P}[t \geq \alpha] = \int_{\alpha}^{+\infty} f(t)dt$$

$$\leq \int_{\alpha}^{+\infty} \frac{t}{\alpha} f(t)dt \text{ (注意 } t \geq \alpha)$$

$$= \frac{\int_{\alpha}^{+\infty} tf(t)dt}{\alpha}$$

$$\leq \frac{\int_{0}^{+\infty} tf(t)dt}{\alpha}$$

$$= \frac{\mathbb{E}(t)}{\alpha}$$

(b)在(a)中取$t = (u - \mu)^2$，注意此处u是随机变量，且

$$\mathbb{E}(u) = \mu, \text{Var}(u) = \mathbb{E}(u - \mu)^2 = \sigma^2$$

所以由(a)

$$\mathbb{P}[(u - \mu)^2 \geq \alpha] \leq \frac{\mathbb{E}((u - \mu)^2)}{\alpha} = \frac{\sigma^2}{\alpha}$$

(c) 由随机变量的性质可得

$$\mathbb{E}(u) = \mathbb{E}(\frac{1}{N} \sum_{n=1}^{N} u_n) = \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}(u_n) = \mu$$

$$\text{Var}(u) = \text{Var}(\frac{1}{N} \sum_{n=1}^{N} u_n) = \frac{1}{N^2} \text{Var}(\sum_{n=1}^{N} u_n) = \frac{1}{N^2} N\sigma^2 = \frac{\sigma^2}{N}$$

第二行的倒数第二个等号利用了独立性。

带入(b)可得

$$\mathbb{P}[(u - \mu)^2 \geq \alpha] \leq \frac{\sigma^2}{N\alpha}$$

Problem 1.9 (Page 37)

In this problem, we derive a form of the law of large numbers that has an exponential bound, called the *Cheroff bound*. We focus on the simple case of flipping a fair coin, and use an approach similar to Problem 1.8.

(a) Let t be a (finite) random variable, α be a positive constant, and s be a positive parameter. If $T(s) = \mathbb{E}_t(e^{st})$, prove that

$$\mathbb{P}[t \geq \alpha] \leq e^{-s\alpha} T(s)$$

[Hint: e^{st} is monotonicaly increasing in t.]

(b) Let $u_1, \cdots, u_N$ be iid random variables, and let $u = \frac{1}{N} \sum_{n=1}^{N} u_n$. If $U(s) = \mathbb{E}_{u_n}(e^{su_n})$(for any n),prove that

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

(c) Suppose $\mathbb{P}[u_n = 0] = \mathbb{P}[u_n = 1] = \frac{1}{2}$ (fair coin). Evaluate $U(s)$ as a function of s, and minimize $e^{-s\alpha} U(s)$ with respect to s for fixed $\alpha, 0 < \alpha < 1$.

(d) Conclude in (c) that, for $0 < \epsilon < \frac{1}{2}$.

$$\mathbb{P}[u \geq \mathbb{E}(u) + \epsilon] \leq 2^{-\beta N}$$

where $\beta = 1 + (\frac{1}{2} + \epsilon) \log_2(\frac{1}{2} + \epsilon) + (\frac{1}{2} - \epsilon) \log_2(\frac{1}{2} - \epsilon)$ and $\mathbb{E}(u) = \frac{1}{2}$. Show that ($\beta > 0$, hence the bound is exponentially decreasing in N.

(a)

$$
\begin{aligned}
\mathbb{P}[t \geq \alpha] &= \int_{\alpha}^{+\infty} f(t)dt \\
&\leq \int_{\alpha}^{+\infty} \frac{e^{st}}{e^{s\alpha}} f(t)dt \text{ (注意 } t \geq \alpha, s > 0) \\
&= e^{-s\alpha} \int_{\alpha}^{+\infty} e^{st} f(t)dt \\
&\leq e^{-s\alpha} \int_{-\infty}^{+\infty} e^{st} f(t)dt \\
&= e^{-s\alpha} \mathbb{E}_t(e^{st}) \\
&= e^{-s\alpha} T(s)
\end{aligned}
$$

(b)

$$
\begin{aligned}
\mathbb{P}[u \geq \alpha] &= \mathbb{P}(\frac{1}{N} \sum_{n=1}^{N} u_n \geq \alpha) \\
&= \mathbb{P}(\sum_{n=1}^{N} u_n \geq N\alpha) \\
&\leq e^{-sN\alpha} \mathbb{E}(e^{s \sum_{n=1}^{N} u_n}) \text{ (由 }(a)) \\
&= e^{-sN\alpha} \prod_{n=1}^{N} E(e^{su_n}) \text{ (由独立性)} \\
&= e^{-sN\alpha}(U(s))^N \\
&= (e^{-s\alpha} U(s))^N
\end{aligned}
$$

(c)由定义知$U(s) = \frac{1}{2}e^s + \frac{1}{2}$，所以$f(s) = e^{-s\alpha} U(s) = \frac{1}{2}(e^{(1-\alpha)s} + e^{-s\alpha})$ 现在需要对固定的α求$U(s)$的最小值

$$f'(s) = \frac{1}{2}[(1-\alpha)e^{(1-\alpha)s} - \alpha e^{-s\alpha}]$$

$$f''(s) = \frac{1}{2}[(1-\alpha)^2 e^{(1-\alpha)s} + \alpha^2 e^{-s\alpha}] > 0$$

所以当$f'(s) = 0$时，$f(s)$取到最小值。令$f'(s) = 0$得

$$\frac{1}{2}[(1-\alpha)e^{(1-\alpha)s} - \alpha e^{-s\alpha}] = 0$$

$$e^s = \frac{\alpha}{1-\alpha}$$

$$s = \ln(\frac{\alpha}{1-\alpha})$$

因此

$$\min f(s) = f(\ln(\frac{\alpha}{1-\alpha}))$$
$$= \frac{1}{2}(\frac{\alpha}{1-\alpha})^{-\alpha}(\frac{\alpha}{1-\alpha} + 1)$$
$$= \frac{1}{2}\frac{\alpha^{-\alpha}}{(1-\alpha)^{-\alpha+1}}$$

因为

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

对任意s均成立，所以

$$\mathbb{P}[u \geq \alpha] \leq \min(e^{-s\alpha}U(s))^N = (\frac{1}{2}\frac{\alpha^{-\alpha}}{(1-\alpha)^{-\alpha+1}})^N$$

(d)$\mathbb{P}[u \geq \mathbb{E}(u) + \epsilon] = \mathbb{P}[u \geq \frac{1}{2} + \epsilon]$，在(c)的不等式中取$\alpha = \frac{1}{2} + \epsilon$得

$$P[u \geq \frac{1}{2} + \epsilon] \leq (\frac{1}{2}\frac{(\frac{1}{2}+\epsilon)^{-(\frac{1}{2}+\epsilon)}}{(\frac{1}{2}-\epsilon)^{\frac{1}{2}-\epsilon}})^N$$
$$= 2^{(-1-(\frac{1}{2}+\epsilon)\log_2(\frac{1}{2}+\epsilon)-(\frac{1}{2}-\epsilon)\log_2(\frac{1}{2}-\epsilon))N}$$
$$= 2^{-\beta N}$$

下面证明

$$\beta = \beta(\epsilon) = 1 + (\frac{1}{2}+\epsilon)\log_2(\frac{1}{2}+\epsilon) + (\frac{1}{2}-\epsilon)\log_2(\frac{1}{2}-\epsilon) > 0$$

求导可得

$$\frac{d\beta}{\epsilon} = \log_2(\frac{1}{2}+\epsilon) + 1 - \log_2(\frac{1}{2}-\epsilon) - 1$$
$$= \log_2\frac{\frac{1}{2}+\epsilon}{\frac{1}{2}-\epsilon} > 0$$

因此β关于ϵ递增，故$\beta > \beta(0) = 0$

Problem 1.10 (Page 37)

Assume that $\mathcal{X} = \{x_1, x_2, \ldots, x_N, x_{N+1}, \ldots, x_{N+M}\}$ and $\mathcal{Y} = \{-1, +1\}$ with an unknown target function $f : \mathcal{X} \to \mathcal{Y}$. The training data set $\mathcal{D}$ is $(x_1, y_1), \ldots, (x_N, y_N)$. Define the *off-training-set* error of a hypothesis h with respect to f by

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

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

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

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

(b) We say that a target function f can 'generate' $\mathcal{D}$ in a noiseless setting if $y_n = f(x_n)$ for all $(x_n, y_n) \in \mathcal{D}$. For a fixed $\mathcal{D}$ of size N, how many possible $f : \mathcal{X} \to \mathcal{Y}$ can generate $\mathcal{D}$ 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 $E_{\text{off}}(h, f) = \frac{k}{M}$? (d) For a given hypothesis h, if all those f that generate $\mathcal{D}$ in a noiseless setting are equally likely in probability, what is the expected off trainingset error $\mathbb{E}_f[E_{\text{off}}(h, f)]$?

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

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

You have now proved that in a noiseless setting, for a fixed $\mathcal{D}$, if all possible f are equally likely, any two deterministic algorithms are equivalent in terms of the expected off training set error. Similar results can be proved fr more general settings.

(a)由题设知$E_{\text{off}}(h, f) = \frac{1}{M} \sum_{m=1}^{M} [\![(N+m)被 2整除]\!]$
我们知道1到N中被2整除的正整数有$[\frac{N}{2}]$个，$[]$为高斯函数，意思为向下取整 因此

$$E_{\text{off}}(h, f) = \frac{1}{M}[(1到 N+M中被 2整除的数的个数) - (1到 N中被 2整除的数的个数)] = \frac{1}{M}([\frac{N+M}{2}] - [\frac{N}{2}])$$

(b)这题的意思是在训练集$\mathcal{D}$上没有误差，在$\{x_{N+1}, \ldots, x_{N+M}\}$上的取值就任意了，每个点有两种取值，所以一共有$2^M$种可以拟合的$f$

(c)一共有M个点，有k个点和目标函数不同，所以共有C_M^k种

(d)题目的意思应该是每个在训练集上无噪声的f出现的概率是一样的，所以这些f在测试集上每个点出现错误的概率应该是一样的，那么由(c)以及(d)可知

$$\mathbb{E}_f\left[E_{\text{Off}}\left(h, f\right)\right] = \sum_{k=0}^{M} \frac{k}{M} \frac{C_M^k}{2^M}$$

$$= \frac{\sum_{k=0}^{M} k C_M^k}{M 2^M}$$

$$= \frac{\sum_{k=1}^{M} M C_{M-1}^{k-1}}{M 2^M}$$

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

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

注意这里用到了$kC_n^k = nC_{n-1}^{k-1}$

(e)由(d)直接推出

Problem 1.11 (Page 38)

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. [Hint: Consider $y_n = +1$ and $y_n = -1$ separately.]

这里假设有N个样本，那么对于Suptermarket：

$$E_{\text{in}} = \frac{1}{N} \sum_{i=1}^{N} (10 \times [\![h(x_n) = -1, f(x_n) = 1]\!] + 1 \times [\![h(x_n) = 1, f(x_n) = -1]\!])$$

对于CIA：

$$E_{\text{in}} = \frac{1}{N} \sum_{i=1}^{N} (1 \times [\![h(x_n) = -1, f(x_n) = 1]\!] + 1000 \times [\![h(x_n) = 1, f(x_n) = -1]\!])$$

Problem 1.12 (Page 38)

This problem investigates how changing the error measure can change the result of the learning process. You have N data points $y_1 \leq \cdots \leq y_N$ and wish to estimate a 'representative' value.

(a) If your algorithm is to find the hypothesis h that minimizes the in sample sum of squared deviations,

$$E_{\text{in}}(h) = \sum_{n=1}^{N} (h - y_n)^2,$$

then show that your estimate will be the in sample mean,

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

(b) If your algorithm is to find the hypothesis h that minimizes the in samples um of absolute deviations,

$$E_{\text{in}}(h) = \sum_{n=1}^{N} |h - y_n|$$

then show that your estimate will be the in sample median h_{med}. which is any value for which half the data points are at most h_{med} and half the data points are at least h_{med}.

(c) Suppose y_N is perturbed to $y_N + \epsilon$, where $\epsilon \to \infty$. So, the single data point y_N becomes an outlier. What happens to your two estimators h_{mean} and h_{med}?

(a)关于h求导可得

$$E'_{\text{in}}(h) = 2 \sum_{n=1}^{N} (h - y_n)$$

$$E''_{\text{in}}(h) = 2N > 0$$

所以$E_{\text{in}}(h)$在$E'_{\text{in}}(h) = 0$的取极小值，解得

$$h = h_{\text{mean}} = \frac{1}{N} \sum_{n=1}^{N} y_n$$

(b)这题证明一个一般结论：

随机变量x的概率密度函数为$f(x)$,

则使得$F(a) = \int_{-\infty}^{+\infty} |x - a| f(x) dx$达到最小值的$a$满足$\int_{-\infty}^{a} f(x) dx = \frac{1}{2}$,

即a为该分布的中位数x_{med}。

证明：首先

$$F(a) = \int_{-\infty}^{a} (a - x) f(x) dx + \int_{a}^{\infty} (x - a) f(x) dx$$

那么关于a求导得

$$F'(a) = \int_{-\infty}^{a} f(x) dx - \int_{a}^{\infty} f(x) dx$$

$$F''(a) = 2f(a) \geq 0$$

所以$F(a)$在$F'(a) = 0$处取最小值，即在$\int_{-\infty}^{a} f(x) dx = \int_{a}^{\infty} f(x) dx = \frac{1}{2}$处取最小值，这一步是由 $\int_{-\infty}^{a} f(x) dx + \int_{a}^{\infty} f(x) dx = 1$得到的，所以$F(a)$在$a = x_{\text{med}}$处取到最小值(离散分布同理可得)。

回到原题，现在构造一个分布，使得$P(y = y_i) = \frac{1}{N}(i = 1, 2...N)$，所以

$$F(h) = \frac{1}{N}E_{\text{in}}(h) = \frac{1}{N}\sum_{n=1}^{N}|h - y_n|$$

因此由之前结论知道$E_{\text{in}}(h)$在$h = y_{\text{med}}$取最小值。

(c)当y_N被污染为$y_N + \epsilon, (\epsilon \to \infty)$时，由定义可知$h_{\text{mean}} \to \infty$,而$y_{\text{med}}$为中位数，则不会有太大变化，因为元素的相对次序只有y_N变了。

最后作为补充，给出Hoeffding不等式的证明：

Hoeffding不等式的证明

首先给出Hoeffding不等式的结论

$$X_1, \ldots, X_n是独立同分布的随机变量，X_i服从参数为p的伯努利分布，$$
$$即\,\mathbb{P}(X_i = 1) = p, \mathbb{P}(X_i = 0) = 1 - p,\ 那么$$
$$\mathbb{P}(|\overline{X} - \mathbb{E}[\overline{X}]| \geq t) \leq 2e^{-2nt^2}$$
$$其中\,\overline{X} = \frac{1}{n}\sum_{i=1}^{n}X_i, \mathbb{E}[\overline{X}] = \frac{1}{n}\mathbb{E}[\sum_{i=1}^{n}X_i] = p, t > 0$$

下面就来证明这个结论，证明方法主要受到MIT概率课程的启发。

首先给出一个引理

引理1 (马尔可夫不等式)

$$Z为非负随机变量，那么\,\mathbb{P}(Z \geq c) \leq \frac{\mathbb{E}[Z]}{c}$$

这个引理是马尔可夫不等式，后面证明中会用到，这个不等式的证明比较简单，见Problem 1.8。

接着，我们不直接证明结论，而是证明一个类似的结论，如下

引理2

$$X_1, \ldots, X_n是独立同分布的随机变量, \mathbb{P}(X_i = 1) = \frac{1}{2} + a, \mathbb{P}(X_i = -1) = \frac{1}{2} - a$$
$$那么\,\mathbb{P}(\overline{X} - \mathbb{E}[\overline{X}] \geq t) \leq e^{-\frac{nt^2}{2}}$$

可以看到这个结论和我们需要证明的结论形式非常类似，但是相对于原来的命题，这个结论更加"对称"一些，这是因为$-1, +1$以及$\frac{1}{2} + a, \frac{1}{2} - a$都比较对称，后面证明中可以看到，这样的对称性可以使得证明更加方便，下面来证明这个结论。

证明：首先计算$\mathbb{E}[X_i], \mathbb{E}[\overline{X}]$

$$\mathbb{E}[X_i] = (\frac{1}{2} + a) \times 1 + (\frac{1}{2} - a) \times (-1) = 2a$$

$$\mathbb{E}[\overline{X}] = \mathbb{E}[X_i] = 2a$$

所以原不等式可以转化为

$$\mathbb{P}(\overline{X} - 2a \geq t) \leq e^{-\frac{nt^2}{2}}$$

以及有如下等价关系

$$\mathbb{P}(\overline{X} - 2a \geq t) \leq e^{-\frac{nt^2}{2}} \Leftrightarrow$$

$$\mathbb{P}(\sum_{i=1}^{n} X_i \geq n(t + 2a)) \leq e^{-\frac{nt^2}{2}} \Leftrightarrow$$

$$\mathbb{P}(e^{s \sum_{i=1}^{n} X_i} \geq e^{sn(t+2a)}) \leq e^{-\frac{nt^2}{2}} \ (s > 0)$$

这里 s 是任意正数，接下来使用引理1

$$\mathbb{P}(e^{s \sum_{i=1}^{n} X_i} \geq e^{sn(t+2a)}) \leq \frac{\mathbb{E}[e^{s \sum_{i=1}^{n} X_i}]}{e^{sn(t+2a)}}$$

我们现在对 $\frac{\mathbb{E}[e^{s \sum_{i=1}^{n} X_i}]}{e^{sn(t+2a)}}$ 进行处理，注意 $X_1, \dots, X_n$ 独立同分布

$$\frac{\mathbb{E}[e^{s \sum_{i=1}^{n} X_i}]}{e^{sn(t+2a)}} = \frac{1}{e^{snt}} \times \frac{(\mathbb{E}[e^{sX_1}])^n}{e^{2asn}} = \frac{1}{e^{snt}} \times (\frac{\mathbb{E}[e^{sX_1}]}{e^{2as}})^n$$

接下来我们处理 $\frac{\mathbb{E}[e^{sX_1}]}{e^{2as}}$，利用 $\mathbb{P}(X_i = 1) = \frac{1}{2} + a, \mathbb{P}(X_i = -1) = \frac{1}{2} - a$

$$\frac{\mathbb{E}[e^{sX_1}]}{e^{2as}} = \frac{e^s(\frac{1}{2} + a) + e^{-s}(\frac{1}{2} - a)}{e^{2as}} = \frac{\frac{1}{2}(e^s + e^{-s}) + a(e^s - e^{-s})}{e^{2as}}$$

记 $m = \frac{1}{2}(e^s + e^{-s}), n = e^s - e^{-s}$，所以上式可以改写为

$$f(a) = \frac{m + na}{e^{2as}}$$

对其取对数可得

$$g(a) = \ln f(a) = \ln(m + na) - 2as$$

研究 $f(a)$ 的极值只要研究 $g(a)$ 的极值即可

$$g'(a) = \frac{n}{m + na} - 2s = 0$$

$$a = \frac{n - 2ms}{2ns}$$

$$g''(a) = -\frac{n^2}{m + na} < 0$$

所以当$a = \frac{n-2ms}{2ns}$时，$g(a)$取极大值，并且$a \leq \frac{n-2ms}{2ns}$时单调递增，$a > \frac{n-2ms}{2ns}$时单调递减，但是注意这里的 $a \in [0, \frac{1}{2}]$，所以还要看$\frac{n-2ms}{2ns}$与$[0, \frac{1}{2}]$的关系，我们先判断$\frac{n-2ms}{2ns}$是否大于0，因为$s > 0$，所以分母 $2ns = 2s(e^s - e^{-s}) > 0$，只要考虑分子即可

$$h(s) = n - 2ms = e^s - e^{-s} - s(e^s + e^{-s})$$
$$h^{'}(s) = e^s + e^{-s} - (e^s + e^{-s}) - s(e^s - e^{-s}) = -s(e^s - e^{-s}) < 0$$
$$h(s) = n - 2ms < h(0) = 0$$

所以$\frac{n-2ms}{2ns} < 0$，从而$g(a)$在$[0, \frac{1}{2}]$上单调递减，因此

$$g(a) \leq g(0)$$
$$f(a) \leq f(0) = m = \frac{1}{2}(e^s + e^{-s})$$

所以现在只要处理$\frac{1}{2}(e^s + e^{-s})$即可，对$e^s, e^{-s}$分别使用泰勒展开

$$e^s = \sum_{i=0}^{+\infty} \frac{s^i}{i!}, e^{-s} = \sum_{i=0}^{+\infty} \frac{(-s)^i}{i!}$$
$$\frac{1}{2}(e^s + e^{-s}) = \frac{1}{2} \sum_{i=0}^{+\infty} \frac{(1 + (-1)^i)}{i!} s^i = \sum_{k=0}^{+\infty} \frac{s^{2k}}{(2k)!}$$

对$(2k)!$稍作变形

$$(2k)! = 1 \times 2 \times \ldots \times k \times (k+1) \times \ldots \times 2k \geq k! \times \underbrace{2 \times \ldots \times 2}_{k \text{个} 2} = 2^k k!$$

将这个式子带入原式可得

$$\frac{1}{2}(e^s + e^{-s}) = \sum_{k=0}^{+\infty} \frac{s^{2k}}{(2k)!} \leq \sum_{k=0}^{+\infty} \frac{(s^2)^k}{k! 2^k} = \sum_{k=0}^{+\infty} \frac{(\frac{s^2}{2})^k}{k!} = e^{\frac{s^2}{2}}$$

把以上几点结合起来可以得到

$$\frac{\mathbb{E}[e^{sX_1}]}{e^{2as}} \leq e^{\frac{s^2}{2}}$$
$$\frac{\mathbb{E}[e^{s \sum_{i=1}^n X_i}]}{e^{sn(t+2a)}} = \frac{1}{e^{snt}} \times \left(\frac{\mathbb{E}[e^{sX_1}]}{e^{2as}}\right)^n \leq \frac{1}{e^{snt}} \times e^{\frac{ns^2}{2}} = (e^{\frac{s^2}{2} - st})^n$$

由于s为任意大于0的数，取$s = t$，从而

$$\frac{\mathbb{E}[e^{s \sum_{i=1}^n X_i}]}{e^{sn(t+2a)}} \leq (e^{-\frac{t^2}{2}})^n = e^{\frac{-nt^2}{2}}$$

$$\mathbb{P}(\overline{X} - \mathbb{E}[\overline{X}] \geq t) = \mathbb{P}(e^{s \sum_{i=1}^n X_i} \geq e^{sn(t+2a)}) \leq \frac{\mathbb{E}[e^{s \sum_{i=1}^n X_i}]}{e^{sn(t+2a)}} \leq e^{-\frac{nt^2}{2}}$$

所以结论得证。这里再补充一点，我们还有以下对称的结论

$$\mathbb{P}(\overline{X} - \mathbb{E}[\overline{X}] \leq -t) \leq e^{-\frac{nt^2}{2}}$$

这是因为

$$\mathbb{P}(\overline{X} - \mathbb{E}[\overline{X}] \leq -t) = \mathbb{P}(-\overline{X} - \mathbb{E}[-\overline{X}] \geq t)$$

因为 $\mathbb{P}(X_i = 1) = \frac{1}{2} + a, \mathbb{P}(X_i = -1) = \frac{1}{2} - a$，所以 $-X_i$ 也是形式一致的随机变量，由引理2可知

$$\mathbb{P}(\overline{X} - E[\overline{X}] \leq -t) = \mathbb{P}(-\overline{X} - E[-\overline{X}] \geq t) \leq e^{-\frac{nt^2}{2}}$$

把以上两者结合有以下推论

$$\mathbb{P}(\left|\overline{X} - \mathbb{E}[\overline{X}]\right| \geq t) = \mathbb{P}(\overline{X} - \mathbb{E}[\overline{X}] \leq -t) + \mathbb{P}(\overline{X} - \mathbb{E}[\overline{X}] \geq t) \leq 2e^{-\frac{nt^2}{2}}$$

最后就利用上述引理2及其推论证明Hoeffding不等式

Hoeffding不等式的证明

Hoeffding不等式中的随机变量 $X_1, \ldots, X_n$ 满足 $\mathbb{P}(X_i = 1) = p, \mathbb{P}(X_i = 0) = 1 - p$，对其稍作变形，转化为引理2的形式

$$Y_i = 2X_i - 1$$
$$\mathbb{P}(Y_i = 1) = p, \mathbb{P}(Y_i = -1) = 1 - p$$

从而

$$\overline{Y} = 2\overline{X} - 1, \mathbb{E}[\overline{Y}] = 2\mathbb{E}[\overline{X}] - 1$$

所以

$$\mathbb{P}(\left|\overline{X} - \mathbb{E}[\overline{X}]\right| \geq t) = \mathbb{P}(\left|2\overline{X} - 2\mathbb{E}[\overline{X}]\right| \geq 2t)$$
$$= \mathbb{P}(\left|2\overline{X} - 1 - (2\mathbb{E}[\overline{X}] - 1)\right| \geq 2t)$$
$$= \mathbb{P}(\left|\overline{Y} - \mathbb{E}[\overline{Y}]\right| \geq 2t)$$

由引理2的推论可知可知

$$\mathbb{P}(\left|\overline{Y} - \mathbb{E}[\overline{Y}]\right| \geq 2t) \leq 2e^{-\frac{n(2t)^2}{2}} = 2e^{-2nt^2}$$

从而

$$\mathbb{P}(\left|\overline{X} - \mathbb{E}[\overline{X}]\right| \geq t) = \mathbb{P}(\left|\overline{Y} - \mathbb{E}[\overline{Y}]\right| \geq 2t) \leq 2e^{-2nt^2}$$

从而结论得证。

总结

这个证明是我自己思考的，可能不是太简洁，但是比较初等，不需要随机过程等知识，证明思路主要是受到MIT概率课程的启发，我个人也强烈推荐这门课。

参考资料：

https://en.wikipedia.org/wiki/Hoeffding%27s_inequality

http://www.xuetangx.com/courses/course-v1:MITx+6_041x+2018_T1/about