

Problem 10

We can observe that the distribution of the update times of the weight w is concentrated around 100. Most of them are in the interval $[95, 105]$, which is about $\frac{N}{2}$. By the figure 1, we can find out that the update times is about half of the numbers of the data.

```

rcv1_train.binary  Hw1P10.py X  plotP10.py  Hw1P12.p
Hw1 > Hw1P10.py > ...
1  import numpy as np
2  from sklearn.datasets import load_svmlight_file
3  import random
4  import matplotlib.pyplot as plt
5
6  X,y = load_svmlight_file("rcv1_train.binary")
7
8  X = X.toarray()
9  X = np.insert(X, 0, values=[1], axis=1)
10
11 xlength, ylength = X.shape
12
13 update = []
14 norm_w_i = []
15 for i in range(1000):
16     N = 0
17     w = np.zeros(ylength)
18     updatetimes = 0
19     norm_w = []
20     while N < 1000:
21         n = random.randrange(0, 199)
22         x = np.array(X[n,:])
23         h = w.T.dot(x)
24         if np.sign(h) == 0:
25             sig = -1
26         else:
27             sig = np.sign(h)
28             if sig != y[n]:
29                 w += y[n]*x
30                 norm_w.append(np.linalg.norm(w))
31                 updatetimes += 1
32                 N = 0
33             else:
34                 N += 1
35     print(i)
36     update.append(updatetimes)
37     norm_w_i.append(norm_w)
38 print(norm_w_i)
39
40 print(update)
41
42 plt.figure(1)
43 T = min(update)
44 for i in range(1000):
45     plt.plot(norm_w_i[i][0:T-1])
46 plt.xlabel("t")
47 plt.ylabel("Norm of w_t")
48
49 plt.figure(2)
50 plt.hist(update, bins=4)
51 plt.xlabel("Update times")
52 plt.ylabel("The numbers of the update times occurs")
53 plt.show()

```

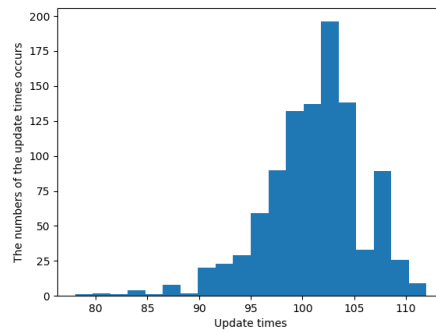


Figure 2: histogram

Figure 1: snapshot

Problem 11

```

rcv1_train.binary  Hw1P10.py x  plotP10.py  Hw1P12.py
Hw1 > Hw1P10.py > ...
1 import numpy as np
2 from sklearn.datasets import load_svmlight_file
3 import random
4 import matplotlib.pyplot as plt
5
6 X,y = load_svmlight_file("rcv1_train.binary")
7
8 X = X.toarray()
9 X = np.insert(X, 0, values=[1], axis=1)
10
11 xlength, ylength = X.shape
12
13 update = []
14 norm_w_i = []
15 for i in range(1000):
16     N = 0
17     w = np.zeros(ylength)
18     updatetimes = 0
19     norm_w = []
20     while N < 1000:
21         n = random.randrange(0, 199)
22         x = np.array(X[n,:])
23         h = w.T.dot(x)
24         if np.sign(h) == 0:
25             sig = -1
26         else:
27             sig = np.sign(h)
28         if sig != y[n]:
29             w += y[n]*x
30             norm_w.append(np.linalg.norm(w))
31             updatetimes += 1
32             N = 0
33         else:
34             N += 1
35     print(i)
36     update.append(updatetimes)
37     norm_w_i.append(norm_w)
38 print(norm_w_i)
39
40 print(update)
41
42 plt.figure(1)
43 T = min(update)
44 for i in range(1000):
45     plt.plot(norm_w_i[i][0:T-1])
46 plt.xlabel("t")
47 plt.ylabel("Norm of w_t")
48
49 plt.figure(2)
50 plt.hist(update, bins=4)
51 plt.xlabel("Update times")
52 plt.ylabel("The numbers of the update times occurs")
53 plt.show()
54

```

Figure 3: snapshot

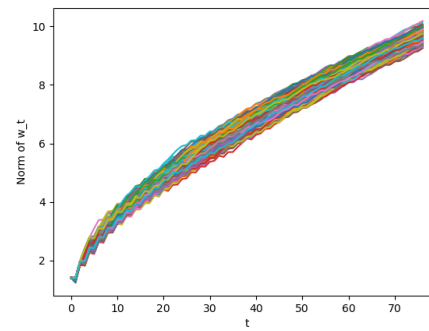


Figure 4: plot

Problem 12

For the P12, we update $n(t)$ only when w_t does not change. Compare with Figure 2 we can observe that most of the update times are still in $[95, 105]$. But it looks more concentrated then the histogram in P10. That is, the update times are more stable than P10.

```

rcv1_train.binary  Hw1P10.py  plotP10.py  Hw1P12.py >
Hw1 > Hw1P12.py > ...
1  import numpy as np
2  from sklearn.datasets import load_svmlight_file
3  import random
4  import matplotlib.pyplot as plt
5
6  X,y = load_svmlight_file("rcv1_train.binary")
7
8  X = X.toarray()
9  X = np.insert(X, 0, values=[1], axis=1)
10
11 xlength, ylength = X.shape
12
13 update = []
14 norm_w_i = []
15 for i in range(1000):
16     N = 0
17     w = np.zeros(ylength)
18     updatetimes = 0
19     norm_w = []
20     n = random.randrange(0, 199)
21     while N < 1000:
22         x = np.array(X[n,:])
23         h = w.T.dot(x)
24         if np.sign(h) == 0:
25             sig = -1
26         else:
27             sig = np.sign(h)
28         if sig != y[n]:
29             w += y[n]*x
30             norm_w.append(np.linalg.norm(w))
31             updatetimes += 1
32             N = 0
33         else:
34             N += 1
35         n = random.randrange(0, 199)
36     print(i)
37     update.append(updatetimes)
38     norm_w_i.append(norm_w)
39 print(norm_w_i)
40
41 print(update)
42
43 # plt.figure(1)
44 # T = min(update)
45 # for i in range(1000):
46 #     plt.plot(norm_w_i[i][0:T-1])
47 # plt.xlabel("t")
48 # plt.ylabel("Norm of w_t")
49 # plt.figure(2)
50 plt.hist(update, bins=4)
51 plt.xlabel("Update times")
52 plt.ylabel("The numbers of the update times occurs")
53 plt.show()

```

Figure 5: snapshot

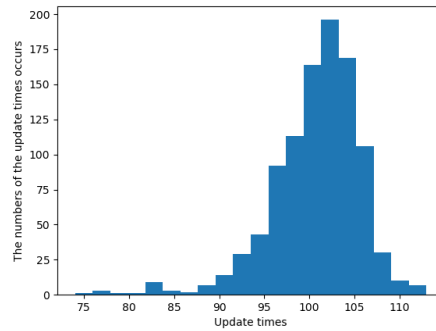


Figure 6: histogram