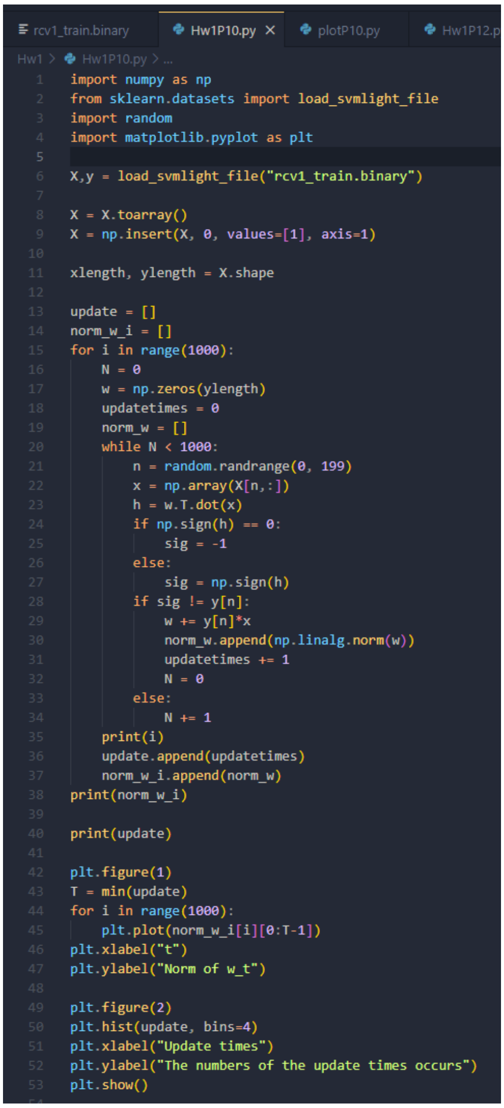
Problem 10

We can observe that the distribution of the update times of the weight $\boldsymbol{w}$ 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.



```python
import numpy as np
from sklearn.datasets import load_svmlight_file
import random
import matplotlib.pyplot as plt

X,y = load_svmlight_file("rcv1_train.binary")

X = X.toarray()
X = np.insert(X, 0, values=[1], axis=1)

xlength, ylength = X.shape

update = []
norm_w_i = []
for i in range(1000):
    N = 0
    w = np.zeros(ylength)
    updatetimes = 0
    norm_w = []
    while N < 1000:
        n = random.randrange(0, 199)
        x = np.array(X[n,:])
        h = w.T.dot(x)
        if np.sign(h) == 0:
            sig = -1
        else:
            sig = np.sign(h)
        if sig != y[n]:
            w += y[n]*x
            norm_w.append(np.linalg.norm(w))
            updatetimes += 1
            N = 0
        else:
            N += 1
    print(i)
    update.append(updatetimes)
    norm_w_i.append(norm_w)
print(norm_w_i)

print(update)

plt.figure(1)
T = min(update)
for i in range(1000):
    plt.plot(norm_w_i[i][0:T-1])
plt.xlabel("t")
plt.ylabel("Norm of w_t")

plt.figure(2)
plt.hist(update, bins=4)
plt.xlabel("Update times")
plt.ylabel("The numbers of the update times occurs")
plt.show()
```

Figure 1: snapshot

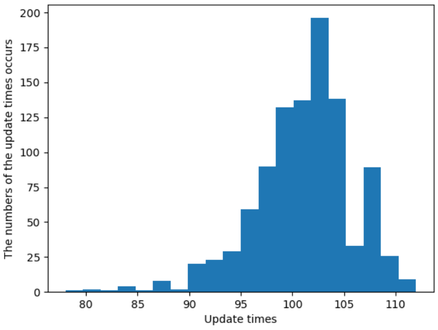
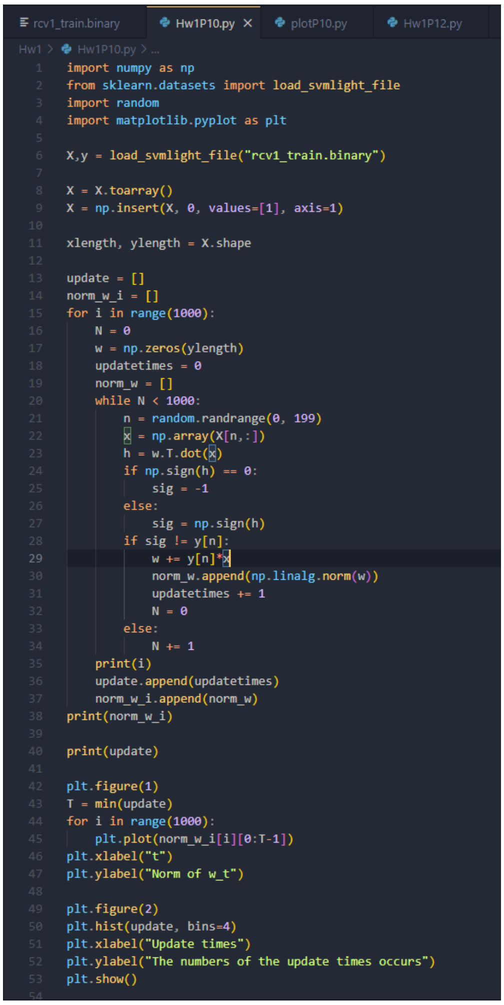


Figure 2: historgram

Problem 11



```python
import numpy as np
from sklearn.datasets import load_svmlight_file
import random
import matplotlib.pyplot as plt

X,y = load_svmlight_file("rcv1_train.binary")

X = X.toarray()
X = np.insert(X, 0, values=[1], axis=1)

xlength, ylength = X.shape

update = []
norm_w_i = []
for i in range(1000):
    N = 0
    w = np.zeros(ylength)
    updatetimes = 0
    norm_w = []
    while N < 1000:
        n = random.randrange(0, 199)
        x = np.array(X[n,:])
        h = w.T.dot(x)
        if np.sign(h) == 0:
            sig = -1
        else:
            sig = np.sign(h)
        if sig != y[n]:
            w += y[n]*x
            norm_w.append(np.linalg.norm(w))
            updatetimes += 1
            N = 0
        else:
            N += 1
    print(i)
    update.append(updatetimes)
    norm_w_i.append(norm_w)
print(norm_w_i)

print(update)

plt.figure(1)
T = min(update)
for i in range(1000):
    plt.plot(norm_w_i[i][0:T-1])
plt.xlabel("t")
plt.ylabel("Norm of w_t")

plt.figure(2)
plt.hist(update, bins=4)
plt.xlabel("Update times")
plt.ylabel("The numbers of the update times occurs")
plt.show()
```

Figure 3: snapshot

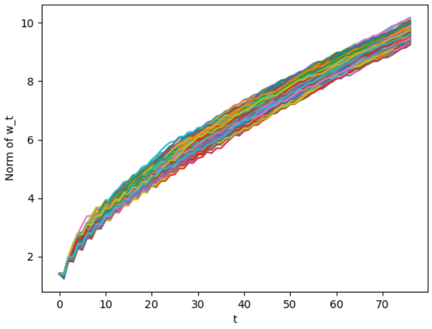
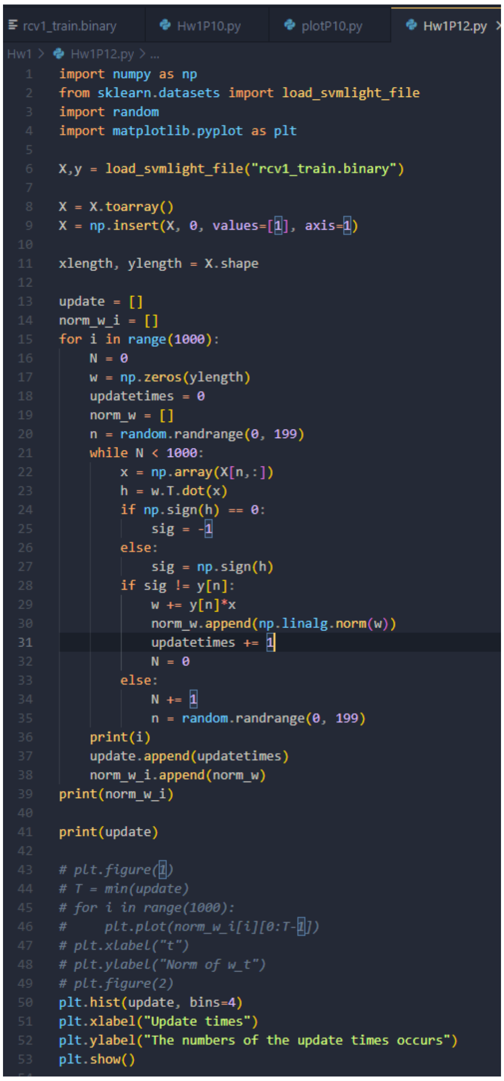


Figure 4: plot

Problem 12

For the P12, we update $n(t)$ only when $\boldsymbol{w}_t$ 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.



```python
import numpy as np
from sklearn.datasets import load_svmlight_file
import random
import matplotlib.pyplot as plt

X,y = load_svmlight_file("rcv1_train.binary")

X = X.toarray()
X = np.insert(X, 0, values=[1], axis=1)

xlength, ylength = X.shape

update = []
norm_w_i = []
for i in range(1000):
    N = 0
    w = np.zeros(ylength)
    updatetimes = 0
    norm_w = []
    n = random.randrange(0, 199)
    while N < 1000:
        x = np.array(X[n,:])
        h = w.T.dot(x)
        if np.sign(h) == 0:
            sig = -1
        else:
            sig = np.sign(h)
        if sig != y[n]:
            w += y[n]*x
            norm_w.append(np.linalg.norm(w))
            updatetimes += 1
            N = 0
        else:
            N += 1
            n = random.randrange(0, 199)
    print(i)
    update.append(updatetimes)
    norm_w_i.append(norm_w)
print(norm_w_i)

print(update)

# plt.figure(1)
# T = min(update)
# for i in range(1000):
#     plt.plot(norm_w_i[i][0:T-1])
# plt.xlabel("t")
# plt.ylabel("Norm of w_t")
# plt.figure(2)
plt.hist(update, bins=4)
plt.xlabel("Update times")
plt.ylabel("The numbers of the update times occurs")
plt.show()
```

Figure 5: snapshot

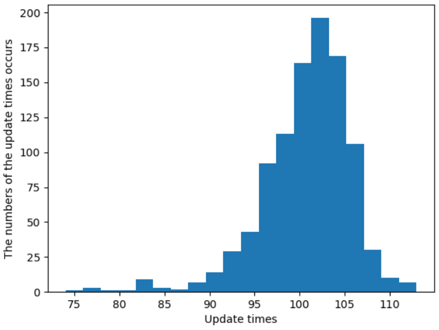


Figure 6: historgram