

2016302580320-任思远-第四次作业

3.5

交替对 w_1, w_2 的样本训练 100 次。

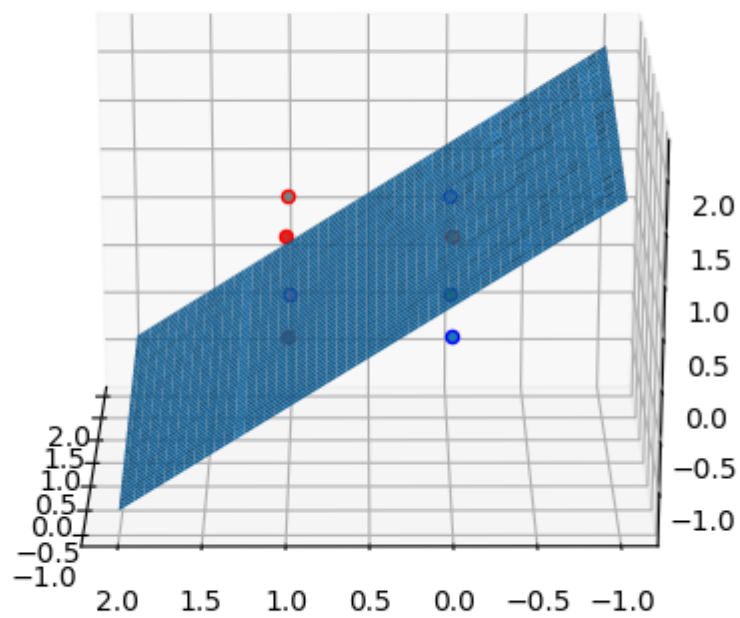
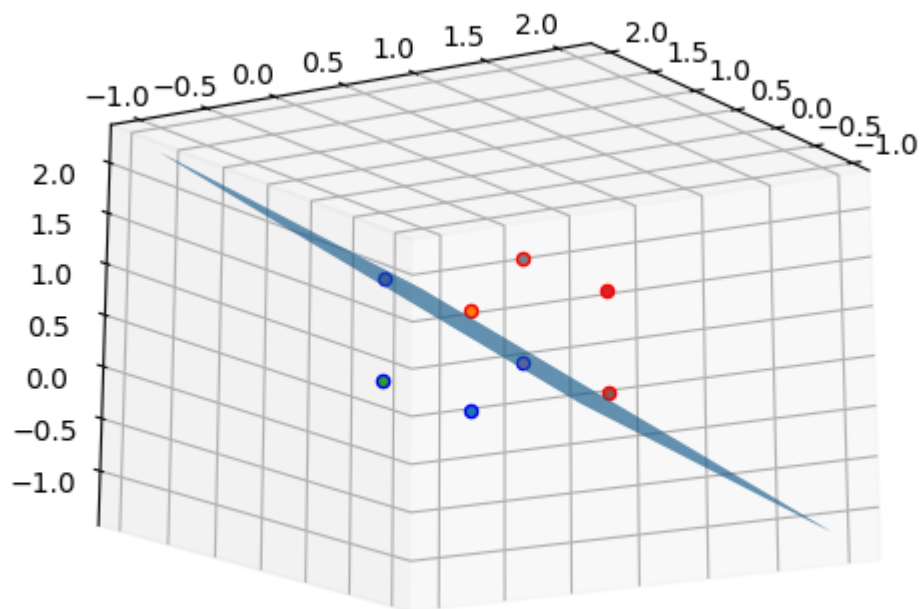
代码: [4_1.py](#)

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from mpl_toolkits.mplot3d import Axes3D
4
5 w1 = np.array([[0, 0, 0, 1],
6                [1, 0, 0, 1],
7                [1, 0, 1, 1],
8                [1, 1, 0, 1]])
9 w2 = np.array([[0, 0, 1, 1],
10               [0, 1, 1, 1],
11               [0, 1, 0, 1],
12               [1, 1, 1, 1]])
13 w2 = w2 * -1
14 W = np.array([-1, -2, -2, 0])
15 c = 0.1
16
17 for times in range(100):
18     for i in range(4):
```

```

19         if W.T.dot(w1[i]) <= 0:
20             W = W + c * w1[i]
21         if W.T.dot(w2[i]) <= 0:
22             W = W + c * w2[i]
23
24     print(W)
25     for i in range(4):
26         print(W.T.dot(w1[i]))
27         print(W.T.dot(w2[i]))
28
29     fig = plt.figure()
30     ax = fig.add_subplot(1, 1, 1, projection='3d')
31     for i in range(4):
32         ax.scatter(w1[i, 0], w1[i, 1], w1[i, 2],
33                   edgecolors='b')
34         ax.scatter(-w2[i, 0], -w2[i, 1], -w2[i, 2],
35                   edgecolors='r') # anti-normalization
36     x = np.linspace(-1, 2, 100)
37     y = np.linspace(-1, 2, 100)
38     X, Y = np.meshgrid(x, y)
39     Z = (-W[3] - W[0] * X - W[1] * Y) / W[2]
40     ax.plot_surface(X, Y, Z)
41     plt.show()

```



3.9

$$K(\mathbf{X}, \mathbf{X}_k) = 1 + (4\mathbf{X}_{(2)}^2 - 2)(4\mathbf{X}_{k(2)}^2 - 2) + (4\mathbf{X}_{(1)}^2 - 2)(4\mathbf{X}_{k(1)}^2 - 2)$$

$$\mathbf{X}_1 = \{0, 1\}$$

$$\mathbf{X}_2 = \{0, -1\}$$

$$\mathbf{X}_3 = \{1, 0\}$$

$$\mathbf{X}_4 = \{-1, 0\}$$

$$k(\mathbf{X}) = 0$$

iteration :

1. $k(\mathbf{X}_1) = 0 \leq 0$, *update :*

$$k(\mathbf{X}) = k(\mathbf{X}) + K(\mathbf{X}, \mathbf{X}_1) = -8\mathbf{X}_{(1)}^2 + 8\mathbf{X}_{(2)}^2 + 1$$

2. $k(\mathbf{X}_2) = -7 < 0$, *ok.*

3. $k(\mathbf{X}_3) = 9 > 0$, *ok.*

4. $k(\mathbf{X}_4) = -7 < 0$, *ok.*

$$\text{Thus, } k = -8\mathbf{X}_{(1)}^2 + 8\mathbf{X}_{(2)}^2 + 1$$

代码: [4_2.cpp](#)

```
1 #include <vector>
2 #include <functional>
3 #include <iostream>
4 #include <numeric>
5
6 void test()
7 {
```

```
8     const std::vector<std::vector<double>> w1 = { {
0, 1 }, { 0, -1 } };
9     const std::vector<std::vector<double>> w2 = { {
1, 0 }, { -1, 0 } };
10
11     const std::function<double(double)> h[3] = {
12         [](double) { return 1.0; },
13         [](double x) { return 2 * x; },
14         [](double x) { return 4 * x * x - 2; },
15     };
16
17     std::function<double(const std::vector<double>&
X)> phi[3] = {
18         [&h](const std::vector<double>& X) { return
h[0](X[0]) * h[0](X[1]); },
19         [&h](const std::vector<double>& X) { return
h[0](X[0]) * h[2](X[1]); },
20         [&h](const std::vector<double>& X) { return
h[2](X[0]) * h[0](X[1]); },
21     };
22
23     const std::function<double(const
std::vector<double>& X, const std::vector<double>&
Xk)> K =
24         [&phi](const std::vector<double>& X, const
std::vector<double>& Xk)
25     {
26         return std::accumulate(std::begin(phi),
std::end(phi), 0.0,
```

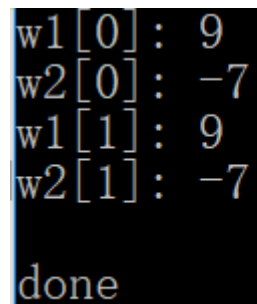
```

27         [&X, &Xk](double val, const
std::function<double(const std::vector<double>& X)>&
phi)
28         { return val + phi(X) * phi(Xk); });
29     };
30
31     std::function<double(const std::vector<double>&
X)> k = [] (const std::vector<double>&) { return 0; };
32
33     for (int times = 0; times < 1; times++)
34         for (int i = 0; i < 2; i++)
35         {
36             if (k(w1[i]) <= 0)
37                 k = [k, i, &w1, &K](const
std::vector<double>& X) { return k(X) + K(X, w1[i]);
};
38             if (k(w2[i]) >= 0)
39                 k = [k, i, &w2, &K](const
std::vector<double>& X) { return k(X) - K(X, w2[i]);
};
40         }
41
42     // check
43     for (int i = 0; i < 2; i++)
44     {
45         std::cout << "w1[" << i << "]: " << k(w1[i])
<< std::endl;
46         std::cout << "w2[" << i << "]: " << k(w2[i])
<< std::endl;
47     }
48

```

```
49 |}  
50  
51 int main()  
52 {  
53     test();  
54     printf("\n%s\n", "done");  
55     getchar();  
56     return 0;  
57 }
```

输出:



```
w1[0]: 9  
w2[0]: -7  
w1[1]: 9  
w2[1]: -7  
  
done
```