

**2020年春季学期  
计算学部《机器学习》课程**

**Lab 2实验报告**

|  |  |
| --- | --- |
| 姓名 | 陈啸 |
| 学号 | 1180300121 |
| 班号 | 1183104 |
| 电子邮件 | [2919730935@qq.com](mailto:2919730935@qq.com) |
| 手机号码 | 13685536606 |

**目录**

[1 实验目的及要求 2](#_Toc27294)

[2 实验环境 2](#_Toc10965)

[3 生成数据 3](#_Toc19735)

[3.1 生成满足朴素贝叶斯假设的数据 3](#_Toc27977)

[3.2 生成不满足满足朴素贝叶斯假设的数据 4](#_Toc10430)

[3.3 结论 4](#_Toc20338)

[4 梯度下降 4](#_Toc2188)

[4.1 代码 4](#_Toc27725)

[4.2 实验结果 5](#_Toc4046)

[5 使用UCI数据集 6](#_Toc1324)

[5.1 选择肿瘤癌数据集 6](#_Toc24803)

[5.2 数据集说明： 6](#_Toc18430)

[5.3 数据集处理 6](#_Toc23258)

# 实验目的及要求

目的：理解逻辑回归模型，掌握逻辑回归模型的参数估计算法。

要求：实现两种损失函数的参数估计（1，无惩罚项；2.加入对参数的惩罚），可以采用梯度下降、共轭梯度或者牛顿法等。

# 实验环境

系统:Win10

Gpu : 1060 630(不知道为啥 使用gpu加速以后依然使用的是集显)

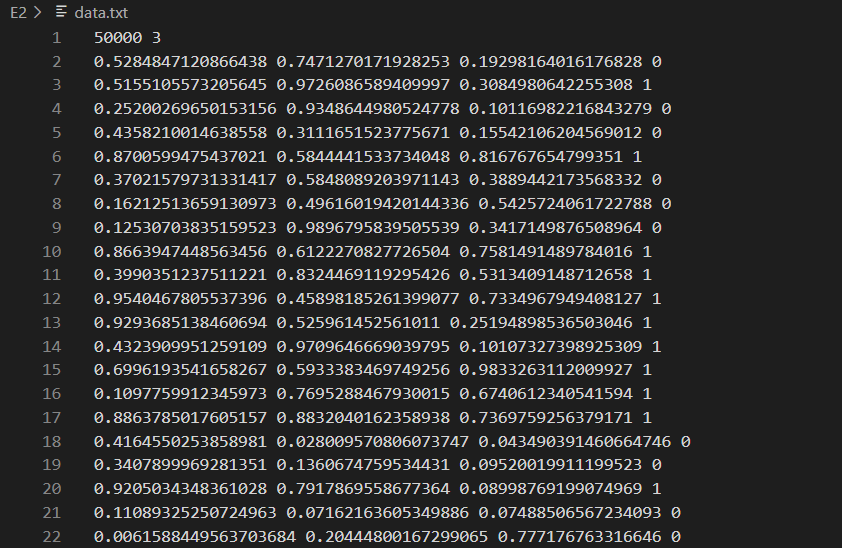
外链库：cupy

# 生成数据

## 生成满足朴素贝叶斯假设的数据

构建一个二分类数据，其中y=（x1+x2+x3）/6<0.6 是就为假，反之为真

1. **import** cupy as cp
3. n = 50000 # 学习集组数
4. y\_div = 0.5
5. d = 3 # 特征值维度
6. m = 10000 #测试集组数
8. f = open("data.txt" , "w")
9. **print** (f)
11. #输出学习集
12. f.write (str(n) + " " + str(d) + "\n")
13. **for** i **in** range (n) :
14. x = cp.random.random (d)
15. **for** j **in** range (d) :
16. f.write (str(x[j]) +' ')
17. **if**(sum (x) < d \* y\_div) :
18. f.write ("0\n")
19. **else**:
20. f.write ("1\n")
22. #输出测试集
23. f.write (str(m) + " " + str(d) + "\n")
24. **for** i **in** range (m) :
25. x = cp.random.random (d)
26. **for** j **in** range (d) :
27. f.write (str(x[j]) +' ')
28. **if**(sum (x) < d \* y\_div) :
29. f.write ("0\n")
30. **else**:
31. f.write ("1\n")
32. f.close()



## 生成不满足满足朴素贝叶斯假设的数据

由于后面在使用uci数据的时候，使用了one-hot，因此该数据集是不满足的，这里不再特别说明。

## 结论

满足与否不影响梯度下降的结果，不过相当影响贝叶斯分类器的结果。

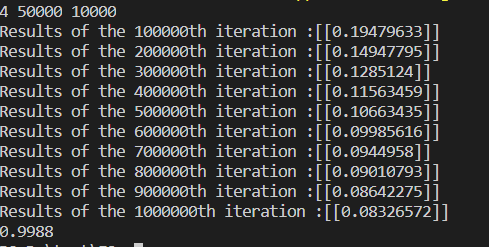
# 梯度下降

## 代码

1. #sigmoid函数
2. **def** g (Z) :
3. **return** 1/(1 + cp.exp(-Z))
5. #损失函数
6. **def** loss(W , X , Y) :
7. y\_hat = g (cp.dot(W.t , X))
8. **return** - (cp.dot (Y.T , cp.log(y\_hat)) + cp.dot( (1 - Y).T , cp.log(1 - y\_hat)))
10. #梯度下降 带正则项
11. **def** gradient\_descent (x , y , w , time=times , mod=100000 ) :
12. # print (mod)
13. # print (time)
14. **for** i **in** range(time) :
15. y\_hat = g (cp.dot(x , w))
16. w = cp.add(w , -1 \* alpha / data\_size \* cp.add (cp.dot(x.T , cp.add(y\_hat , -1 \* y)) , lambd  \* w) )
17. **if** (i+1)%mod==0 :
18. **print** ('Results of the ', end='')
19. **print**(i+1, end='')
20. **print**('th iteration :' , end='')
21. **print**((-1 \* cp.add(cp.dot(y.T , cp.log(y\_hat)) , cp.dot((1-y.T) , cp.log(1-y\_hat))) + lambd / 2 \* cp.dot(w.T ,w)) / data\_size)
22. **return** w
24. #随机初始化
25. **def** random\_initialization (p , q) :
26. w = cp.ndarray(shape=(p , q) , dtype = float)
27. **for** i **in** range(p) :
28. **for** j **in** range (q) :
29. w[i,j] = cp.random.random()\*0.01
30. **return** w
31. #进行梯度下降 其中w初始值为随机初始化的结果
32. w = gradient\_descent (x\_train , y\_train , random\_initialization(M , 1) )

## 实验结果

使用3.1中生成的数据，进行梯度下降的计算，得到结果如下



# 使用UCI数据集

## 选择肿瘤癌数据集

数据集来源：https://archive.ics.uci.edu/ml/datasets/Breast+Cancer

## 数据集说明：

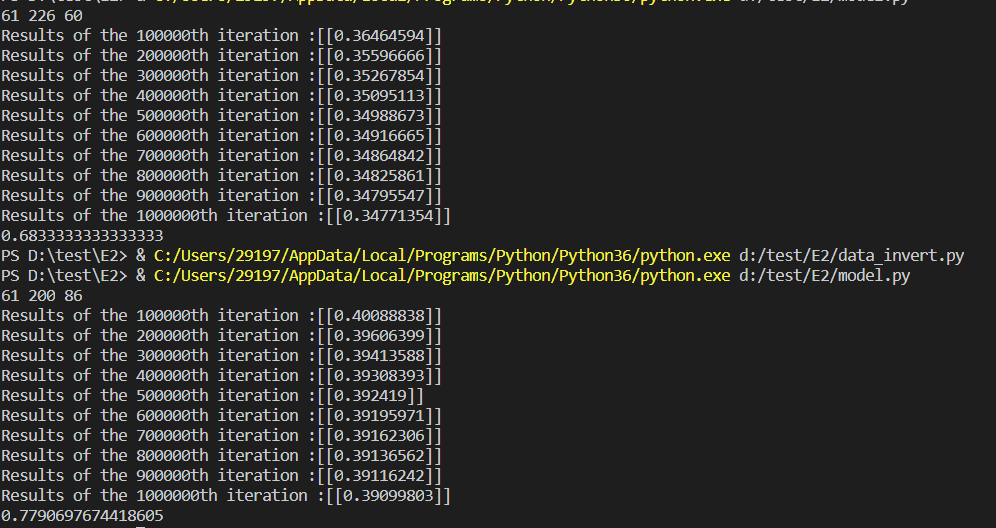
其中每个病人总共有9个属性的离散的特征值，分为年龄，大小等等，其中部分病人的部分特征值缺失，为‘?’

## 数据集处理

因此我们使用one\_hot方法离散所有的特征，其中每个特征的?也作为一个维度，这样总共有60个维度，即一个病人对应一个(1,60)的one\_hot变量

1. **import** cupy as cp
3. doc = ["no-recurrence-events","recurrence-events","?","10-19","20-29",
4. "30-39","40-49","50-59","60-69","70-79",
5. "80-89","90-99","?","lt40","ge40",
6. "premeno","?","0-4","5-9","10-14",
7. "15-19","20-24","25-29","30-34","35-39",
8. "40-44","45-49","50-54","55-59","?",
9. "0-2","3-5","6-8","9-11","12-14",
10. "15-17","18-20","21-23","24-26","27-29",
11. "30-32","33-35","36-39","?","yes",
12. "no","?","1","2","3",
13. "?","left","right","?","left-up",
14. "left-low","right-up","right-low","central","?"]
15. index = [0,0,0,1,1,
16. 1,1,1,1,1,
17. 1,1,1,2,2,
18. 2,2,3,3,3,
19. 3,3,3,3,3,
20. 3,3,3,3,3,
21. 4,4,4,4,4,
22. 4,4,4,4,4,
23. 4,4,4,4,4,
24. 4,4,4,4,5,
25. 5,5,6,6,6,
26. 6,7,7,7,8,
27. 8,8,8,8,8]
29. f = open("breast-cancer.data","r")
30. M = 60
31. test\_size = 60
32. fstr = f.read()
33. fstr = fstr.split('\n')
34. data\_size = len(fstr)
35. data = cp.ndarray(shape = (data\_size , M) , dtype = float )
36. y = cp.ndarray(shape = (data\_size , 1) , dtype = float )
37. **for** i **in** range(data\_size) :
38. **for** j **in** range(M):
39. data[i,j] = 0
40. **for** i **in** range(data\_size) :
41. ss = fstr[i].split(',')
42. **for** j **in** range(M):
43. **if**(ss[index[j]] == doc [j]) :
44. data[i,j]=1
45. **if** ss[9] == "yes" :
46. y[i] = 1
47. **else** :
48. y[0] = 0
49. f.close()
50. cnt = 0
51. vis = cp.ndarray(shape =(data\_size) , dtype = int)
52. **while**(cnt < test\_size) :
53. random\_x = cp.random.randint(0 , data\_size)
54. **if**(vis[random\_x] == 0) :
55. vis[random\_x] = 1
56. cnt = cnt + 1
57. f = open("data.txt","w")
58. f.write(str(data\_size - test\_size)+" "+str(M)+"\n")
59. # f.write(str(data\_size)+" "+str(M)+"\n")
60. **for** i **in** range(data\_size) :
61. if vis[i] == 1 :
62. continue
63. **for** j **in** range(M):
64. f.write(str(data[i,j])+' ')
65. f.write(str(y[i,0])+"\n")
66. f.write(str(test\_size)+" "+str(M)+"\n")
67. **for** i **in** range(data\_size) :
68. **if** vis[i] == 0 :
69. **continue**
70. **for** j **in** range(M):
71. f.write(str(data[i,j])+' ')
72. f.write(str(y[i,0])+"\n")

其中，总共有286个病人，我们选择226个病人样例作为学习集，另外的60个病人数据作为测试集



其中第二组结果使用了更少的病人作为学习集合，正确率上升了。

# 总结

1. 梯度下降算法的最终运行结果与提供的数据集中特征之间的相关性关系不大，满足朴素贝叶斯假设与否都可以正常求解
2. 对于分布较为平均的数据，比如3.1中自己生成的随机数据，其学习集数据组数在超过一定值之后对结果几乎没有影响，迭代次数在超多一定次数之后，loss几乎不变，这意味着超过一定次数之后的迭代结果变化不大，此时可以考虑使用更小的alpha值。
3. 对于分布不平均的数据，比如5.1中从uci获取的肿瘤癌数据集，即使使用了one\_hot离散特征，最后的准确率与学习集大小依然息息相关，更多的学习集应该能得出更好的结果。