**大 数 据 分 析（2019秋）Homework1**

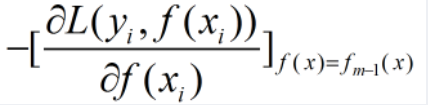
2019.9.20

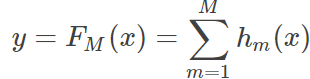
1. 比较说一下 Random Forest 与 GBDT的共同点与区别？

RF算法与GBDT算法都是决策树算法与集成思想相结合产生的，在训练过程中会生成多棵决策树，并整合这些决策树的输出提升整体模型的分类或回归的效果。实践证明他们都有比较好的鲁棒性和泛化能力，在实际应用中由3比较好的表现。

RF算法的训练思想主要基于bagging，该算法最主要的目的就是在决策树的训练过程中引入尽可能多的随机性（不仅随机选择样本、而且随机选择特征，有放回采样），然后使用bagging方法进行整合，主要关注降低方差，其基学习器还是传统的CART决策树，。而GBDT算法的基本思想来自于Boosting，使用梯度上升方法从根节点开始生成多棵决策树（训练样本使用无放回抽样），并且它使用的决策树是回归树，所以更适合回归问题。

在训练方面，RF算法由于使用的基学习器还是CART树，其训练依然使用Gini指数；在训练时需要给定随机训练集D、特征维数F，训练参数需要指定决策树数量n、每棵树深度d、每个结点使用的特征维度k。而GBDT算法使用迭代的方法，使用前向分布算法，假设前一轮迭代的得到的学习器为强学习器，本轮迭代得到的弱学习器的目标函数是使得损失函数最小化；GBDT的学习器使用的是CART回归树，并且目标损失函数使用残差减小的梯度方向（这里使用损失函数的负梯度在当前模型的值作为回归问题中残差或者残差的近似值 ），所以这里不再使用普通Boosting算法中的残差，而是使用损失函数的梯度作为新的训练数据的y值

，

其中，表示上一轮迭代所有回归树的预测值之和作为当前迭代的训练标签数据。

另外，RF算法一般将树的深度设置的比较深，而GBDT算法的深度一般在6左右就能得到比较好的效果。是因为GBDT算法属于boosting算法，因为有很多简单的回归树结构，方差小，在训练是该算法就可以更加专注于降低训练偏差；而RF算法使用更深的更复杂的结构，偏差小，在训练时就可以更加关注降低训练方差。

在预测时，当所有的树建好后，可以并行进行预测，然后将所有树的预测值进行加和作为整体的回归值；而RF算法在预测时也可以使用并行化方法，当解决回归问题时，预测值应该使用所有树的平均值。

2. 写程序利用SVD分解对数据进行降维，并画出降维后各个节点在”新维度“上值的分布。例如对于矩阵 A(mxn) = , U 分解后每一列 有m个元素对应m个点，记为. 我们画出**-**的坐标下的m个点的散点图。

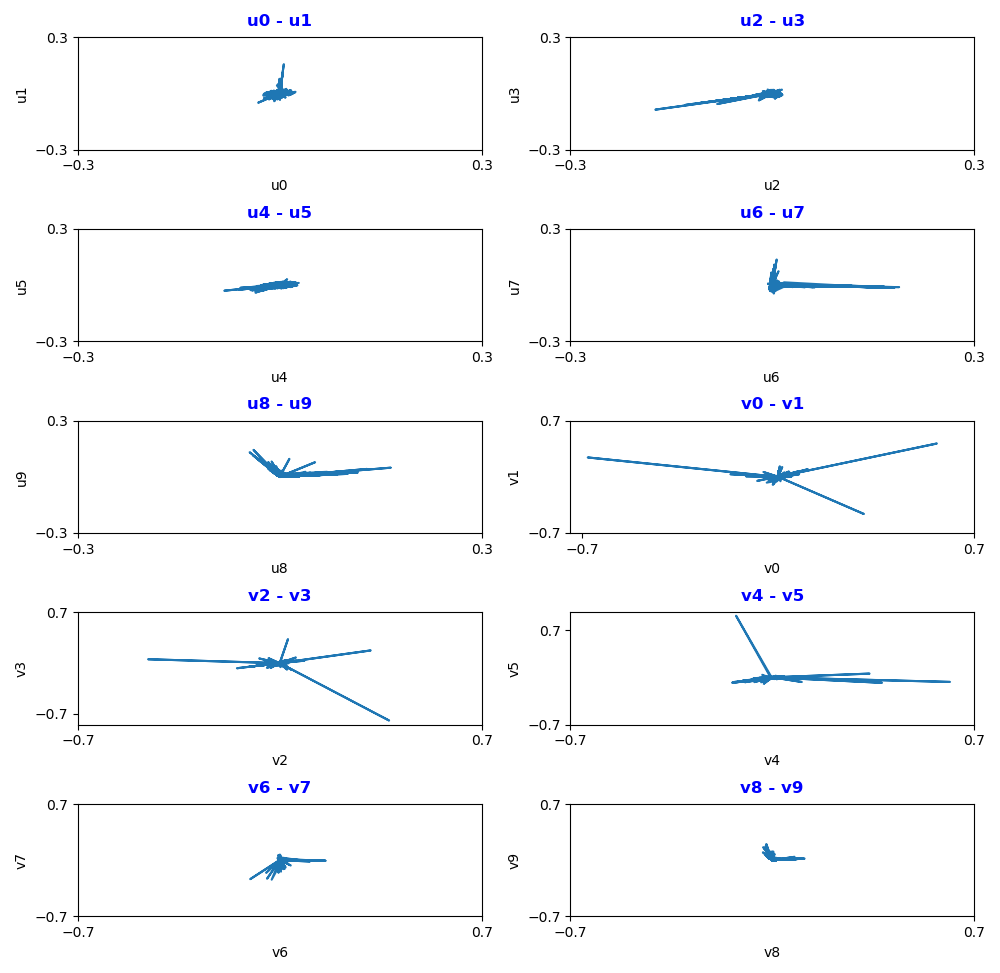


图1 yelp.edgelist.gz数据集SVD分解的Spectral Subspace Plots

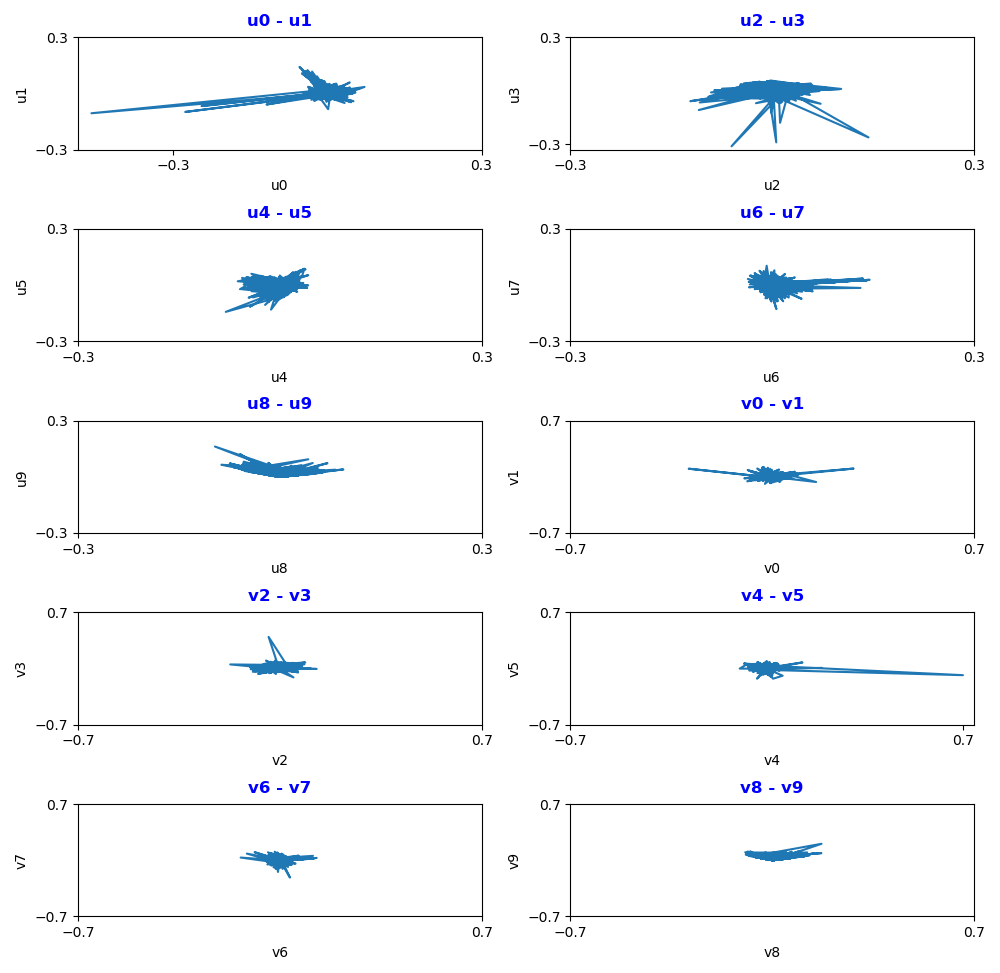


图2 Epinions social network数据集（[soc-Epinions1.txt.gz](http://snap.stanford.edu/data/soc-Epinions1.txt.gz)），

网站链接（<http://snap.stanford.edu/data/soc-Epinions1.html>）

**代码1老师给定数据集的SVD分解**

1. #！user/bin/python
2. # -\*- coding:UTF-8 -\*-
4. '''''
5. file\_name: 大数据分析作业1-1，老师给定数据集的svd降维
6. os: win10
7. python: 3.7
8. author: 从聪, 学号 201928015029015
9. '''
10. **import** os, path
11. **from** scipy.sparse **import** csr\_matrix
12. **import** gzip
13. **import** matplotlib.pyplot as plt
14. **import** scipy.sparse.linalg as slin
15. **import** numpy as np
16. # 直接读取.gz文件
17. **def** load\_file(file\_name, mode):
18. f = gzip.open(file\_name, mode)
19. **return** f
21. # 生成行坐标、列坐标、数据值
22. **def** gen\_csr\_matrix(file\_name):
23. xs = []
24. ys = []
25. data =[]
26. with load\_file(file\_name, 'rb') as file\_data:
27. datas = file\_data.readlines()
28. **for** line **in** datas:
29. line = line.strip().decode().split(' ')
30. xs.append(int(line[0]))
31. ys.append(int(line[1]))
32. data.append(1)
33. file\_data.close()
34. m = max(xs) + 1
35. n = max(ys) + 1
36. # print( m,n,len(data))
37. Mat = csr\_matrix((data, (xs, ys)), shape=(m, n),dtype=np.float64)
38. **return** Mat
39. #画出10副图
40. **def** plt\_u\_gra(U, v):
41. plt.figure(figsize=(10,10))
42. **for** i **in** range(0,10,2):
43. plt.subplot(5, 2, 1 + int(i/2))
44. plt.plot(U[:,i], U[:,i+1])
45. plt.xlabel('u'+str(i))
46. plt.ylabel('u'+str(i+1))
47. plt.title('u'+str(i) + ' - u'+str(i+1), color='blue', fontweight='bold', verticalalignment='bottom')
48. plt.xticks([-0.3,0.3])
49. plt.yticks([-0.3,0.3])
51. **for** j **in** range(0,10,2):
52. plt.subplot(5, 2, 6 + int(j/2))
53. plt.plot(v[j], v[j+1])
54. plt.xlabel('v'+str(j))
55. plt.ylabel('v'+str(j+1))
56. plt.title('v'+str(j) + ' - v'+str(j+1), color='blue', fontweight='bold', verticalalignment='bottom')
57. plt.xticks([-0.7,0.7])
58. plt.yticks([-0.7,0.7])
59. # plt.tight\_layout()
60. plt.tight\_layout()
61. plt.show()
63. **if** \_\_name\_\_ == "\_\_main\_\_":
64. file\_name = "./pros/data/yelp.edgelist.gz"
65. Mat = gen\_csr\_matrix(file\_name)
66. Mat = Mat.asfptype()
67. u, sigma, vt = slin.svds(Mat,k=10,which='LM')
68. # print(np.array(u)[:,0].shape)
69. # print(u.shape)
70. plt\_u\_gra(u, vt)

**代码2 公开数据集的SVD分解**

1. #！user/bin/python
2. # -\*- coding:UTF-8 -\*-
4. '''''
5. file\_name: 大数据分析作业1-2，公开数据集的svd降维
6. task\_name: Epinions social network, url: http://snap.stanford.edu/data/soc-Epinions1.html
7. data\_set\_name: soc-Epinions1.txt.gz, url: http://snap.stanford.edu/data/soc-Epinions1.txt.gz
8. os: win10
9. python: 3.7
10. author: 从聪, 学号 201928015029015
11. '''
13. **import** os, path
14. **from** scipy.sparse **import** csr\_matrix
15. **import** gzip
16. **import** matplotlib.pyplot as plt
17. **import** scipy.sparse.linalg as slin
18. **import** numpy as np
20. # 直接读取.gz文件
21. **def** load\_file(file\_name, mode):
22. f = gzip.open(file\_name, mode)
23. **return** f
25. # 生成行坐标、列坐标、数据值
26. **def** gen\_csr\_matrix(file\_name):
27. xs = []
28. ys = []
29. data =[]
31. with load\_file(file\_name, 'rb') as file\_data:
32. datas = file\_data.readlines()
33. **for** line **in** datas:
34. line = line.strip().decode().split('\r')[0].split('\t')
35. **if** line[0][0]=='' **or** line[0][0]=='#':
36. **continue**
37. xs.append(int(line[0]))
38. ys.append(int(line[1]))
39. data.append(1)
40. file\_data.close()
42. m = max(xs) + 1
43. n = max(ys) + 1
44. **print**( m,n,len(data))
45. Mat = csr\_matrix((data, (xs, ys)), shape=(m, n),dtype=np.float64)
46. **return** Mat
48. # 画原始数据的分布图
49. **def** plt\_ori\_data(Mat):
50. idxs = Mat.todok().tocoo()
51. rows = list(idxs.row.reshape(-1))
52. cols = list(idxs.col.reshape(-1))
53. # print(len(row))
55. plt.plot(rows, cols)
56. ax = plt.gca()
57. ax.xaxis.set\_ticks\_position('top')
58. ax.invert\_yaxis()
59. plt.show()
61. #画出10副图
62. **def** plt\_u\_gra(U, v):
63. plt.figure(figsize=(10,10))
64. **for** i **in** range(0,10,2):
65. plt.subplot(5, 2, 1 + int(i/2))
66. plt.plot(U[:,i], U[:,i+1])
67. plt.xlabel('u'+str(i))
68. plt.ylabel('u'+str(i+1))
69. plt.title('u'+str(i) + ' - u'+str(i+1), color='blue', fontweight='bold', verticalalignment='bottom')
70. plt.xticks([-0.3,0.3])
71. plt.yticks([-0.3,0.3])
73. **for** j **in** range(0,10,2):
74. plt.subplot(5, 2, 6 + int(j/2))
75. plt.plot(v[j], v[j+1])
76. plt.xlabel('v'+str(j))
77. plt.ylabel('v'+str(j+1))
78. plt.title('v'+str(j) + ' - v'+str(j+1), color='blue', fontweight='bold', verticalalignment='bottom')
79. plt.xticks([-0.7,0.7])
80. plt.yticks([-0.7,0.7])
81. # plt.tight\_layout()
82. plt.tight\_layout()
83. plt.show()
85. **if** \_\_name\_\_ == "\_\_main\_\_":
86. file\_name = "./pros/data/task\_2\_soc-Epinions1.txt.gz"
87. Mat = gen\_csr\_matrix(file\_name)
88. **print**(Mat.shape)
89. # plt\_ori\_data(Mat)
90. Mat = Mat.asfptype()
91. u, sigma, vt = slin.svds(Mat,k=10,which='LM')
92. # # print(np.array(u)[:,0].shape)
93. # # print(u.shape)
94. plt\_u\_gra(u, vt)