# 实验报告

17184111班 陶逸群 17042127

## 实验过程

在这次实验中，我采用了支持向量机法对所给数据进行了分类。由于实验数据没有给出特定含义，故将其全部作为连续型数据处理，且实验数据没有缺失，不必进行数据补全，因此后续对实验数据进行了主成分分析，选取出170个差异性较大的主成分，重新进行分类，加快了分类时间，而且提高了分类准确率。

## 参数选择

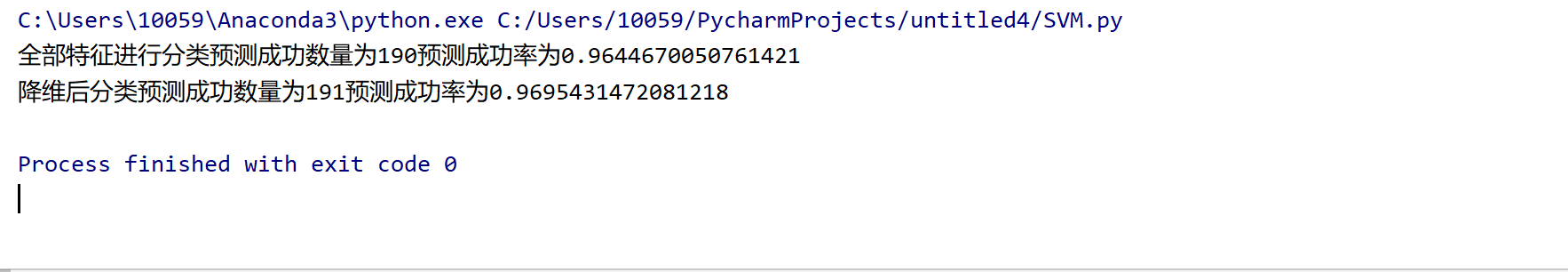
使用支持向量机进行分类，首先要确定分类所采用的内核，一般来说有高斯核和线性核可供选择，而本实验的所给样本数量远远小于特征数量，不需要使用高斯核将其映射到高维空间（容易过拟合），因此采用线性核。经过我的测试，也发现在其他参数都默认的情况下，采用高斯核进行分类的效果远远不如采用线性核分类的效果，故最终选择线性核进行分类。

在采用线性核进行分类的情况下，选取C的值越大，分类效果越好，但是C越大，也有可能发生过拟合现象，导致效果变差。经过实验，发现C选取默认值1时，分类效果达到最佳。

采用主成分分析法进行数据降维，发现当保留170维时，分类效果达到最佳。

## 实验结果

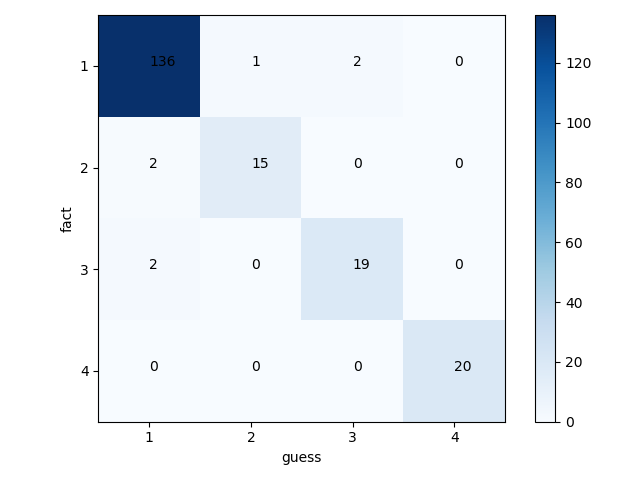
首先在控制台输出

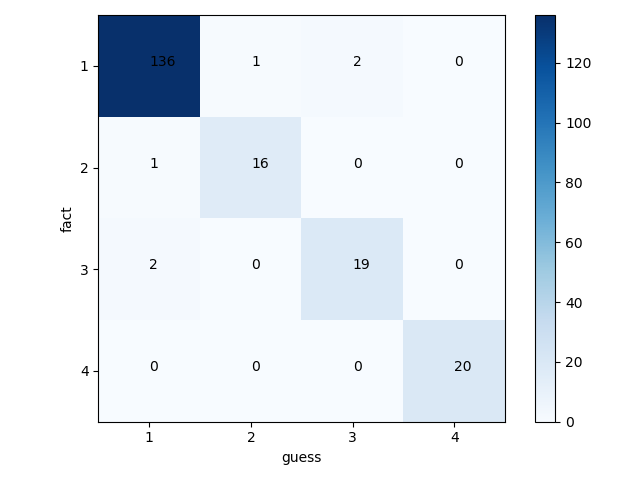


并且输出分类结果文件，其中第一类是原数据，第二列是全部特征预测结果，第三列是降维后预测结果。

|  |  |  |  |
| --- | --- | --- | --- |
|  | **0** | **0** | **0** |
| **0** | 1 | 1 | 1 |
| **1** | 1 | 1 | 1 |
| **2** | 1 | 1 | 1 |
| **3** | 1 | 1 | 1 |
| **4** | 1 | 1 | 1 |
| **5** | 1 | 1 | 1 |
| **6** | 1 | 1 | 1 |
| **7** | 1 | 1 | 1 |
| **8** | 1 | 1 | 1 |
| **9** | 1 | 1 | 1 |
| **10** | 1 | 1 | 1 |
| **11** | 1 | 3 | 3 |
| **12** | 1 | 1 | 1 |
| **13** | 1 | 1 | 1 |
| **14** | 1 | 1 | 1 |
| **15** | 1 | 1 | 1 |
| **16** | 1 | 1 | 1 |
| **17** | 1 | 1 | 1 |
| **18** | 1 | 1 | 1 |
| **19** | 1 | 1 | 1 |
| **20** | 1 | 1 | 1 |
| **21** | 1 | 1 | 1 |
| **22** | 1 | 1 | 1 |
| **23** | 1 | 1 | 1 |
| **24** | 1 | 1 | 1 |
| **25** | 1 | 1 | 1 |
| **26** | 1 | 1 | 1 |
| **27** | 1 | 1 | 1 |
| **28** | 1 | 1 | 1 |
| **29** | 1 | 1 | 1 |
| **30** | 1 | 1 | 1 |
| **31** | 1 | 1 | 1 |
| **32** | 1 | 1 | 1 |
| **33** | 1 | 1 | 1 |
| **34** | 1 | 1 | 1 |
| **35** | 1 | 1 | 1 |
| **36** | 1 | 1 | 1 |
| **37** | 1 | 1 | 1 |
| **38** | 1 | 1 | 1 |
| **39** | 1 | 1 | 1 |
| **40** | 1 | 1 | 1 |
| **41** | 1 | 1 | 1 |
| **42** | 1 | 1 | 1 |
| **43** | 1 | 1 | 1 |
| **44** | 1 | 1 | 1 |
| **45** | 1 | 1 | 1 |
| **46** | 1 | 3 | 3 |
| **47** | 1 | 1 | 1 |
| **48** | 1 | 1 | 1 |
| **49** | 1 | 1 | 1 |
| **50** | 1 | 1 | 1 |
| **51** | 1 | 1 | 1 |
| **52** | 1 | 1 | 1 |
| **53** | 1 | 1 | 1 |
| **54** | 1 | 1 | 1 |
| **55** | 1 | 1 | 1 |
| **56** | 1 | 1 | 1 |
| **57** | 1 | 1 | 1 |
| **58** | 1 | 1 | 1 |
| **59** | 1 | 1 | 1 |
| **60** | 1 | 1 | 1 |
| **61** | 1 | 1 | 1 |
| **62** | 1 | 1 | 1 |
| **63** | 1 | 1 | 1 |
| **64** | 1 | 1 | 1 |
| **65** | 1 | 1 | 1 |
| **66** | 1 | 1 | 1 |
| **67** | 1 | 1 | 1 |
| **68** | 1 | 1 | 1 |
| **69** | 1 | 1 | 1 |
| **70** | 1 | 1 | 1 |
| **71** | 1 | 1 | 1 |
| **72** | 1 | 1 | 1 |
| **73** | 1 | 1 | 1 |
| **74** | 1 | 1 | 1 |
| **75** | 1 | 1 | 1 |
| **76** | 1 | 1 | 1 |
| **77** | 1 | 1 | 1 |
| **78** | 1 | 1 | 1 |
| **79** | 1 | 1 | 1 |
| **80** | 1 | 1 | 1 |
| **81** | 1 | 1 | 1 |
| **82** | 1 | 1 | 1 |
| **83** | 1 | 1 | 1 |
| **84** | 1 | 1 | 1 |
| **85** | 1 | 1 | 1 |
| **86** | 1 | 1 | 1 |
| **87** | 1 | 1 | 1 |
| **88** | 1 | 1 | 1 |
| **89** | 1 | 1 | 1 |
| **90** | 1 | 1 | 1 |
| **91** | 1 | 1 | 1 |
| **92** | 1 | 1 | 1 |
| **93** | 1 | 1 | 1 |
| **94** | 1 | 1 | 1 |
| **95** | 1 | 1 | 1 |
| **96** | 1 | 1 | 1 |
| **97** | 1 | 1 | 1 |
| **98** | 1 | 1 | 1 |
| **99** | 1 | 1 | 1 |
| **100** | 1 | 1 | 1 |
| **101** | 1 | 1 | 1 |
| **102** | 1 | 1 | 1 |
| **103** | 1 | 1 | 1 |
| **104** | 1 | 1 | 1 |
| **105** | 1 | 1 | 1 |
| **106** | 1 | 1 | 1 |
| **107** | 1 | 1 | 1 |
| **108** | 1 | 1 | 1 |
| **109** | 1 | 1 | 1 |
| **110** | 1 | 1 | 1 |
| **111** | 1 | 1 | 1 |
| **112** | 1 | 1 | 1 |
| **113** | 1 | 1 | 1 |
| **114** | 1 | 1 | 1 |
| **115** | 1 | 1 | 1 |
| **116** | 1 | 2 | 2 |
| **117** | 1 | 1 | 1 |
| **118** | 1 | 1 | 1 |
| **119** | 1 | 1 | 1 |
| **120** | 1 | 1 | 1 |
| **121** | 1 | 1 | 1 |
| **122** | 1 | 1 | 1 |
| **123** | 1 | 1 | 1 |
| **124** | 1 | 1 | 1 |
| **125** | 1 | 1 | 1 |
| **126** | 1 | 1 | 1 |
| **127** | 1 | 1 | 1 |
| **128** | 1 | 1 | 1 |
| **129** | 1 | 1 | 1 |
| **130** | 1 | 1 | 1 |
| **131** | 1 | 1 | 1 |
| **132** | 1 | 1 | 1 |
| **133** | 1 | 1 | 1 |
| **134** | 1 | 1 | 1 |
| **135** | 1 | 1 | 1 |
| **136** | 1 | 1 | 1 |
| **137** | 1 | 1 | 1 |
| **138** | 1 | 1 | 1 |
| **139** | 2 | 2 | 2 |
| **140** | 2 | 2 | 2 |
| **141** | 2 | 2 | 2 |
| **142** | 2 | 2 | 2 |
| **143** | 2 | 2 | 2 |
| **144** | 2 | 2 | 2 |
| **145** | 2 | 1 | 1 |
| **146** | 2 | 2 | 2 |
| **147** | 2 | 2 | 2 |
| **148** | 2 | 2 | 2 |
| **149** | 2 | 2 | 2 |
| **150** | 2 | 2 | 2 |
| **151** | 2 | 2 | 2 |
| **152** | 2 | 2 | 2 |
| **153** | 2 | 2 | 2 |
| **154** | 2 | 2 | 2 |
| **155** | 2 | 1 | 2 |
| **156** | 3 | 3 | 3 |
| **157** | 3 | 3 | 3 |
| **158** | 3 | 3 | 3 |
| **159** | 3 | 3 | 3 |
| **160** | 3 | 3 | 3 |
| **161** | 3 | 3 | 3 |
| **162** | 3 | 3 | 3 |
| **163** | 3 | 3 | 3 |
| **164** | 3 | 3 | 3 |
| **165** | 3 | 3 | 3 |
| **166** | 3 | 1 | 1 |
| **167** | 3 | 3 | 3 |
| **168** | 3 | 3 | 3 |
| **169** | 3 | 1 | 1 |
| **170** | 3 | 3 | 3 |
| **171** | 3 | 3 | 3 |
| **172** | 3 | 3 | 3 |
| **173** | 3 | 3 | 3 |
| **174** | 3 | 3 | 3 |
| **175** | 3 | 3 | 3 |
| **176** | 3 | 3 | 3 |
| **177** | 4 | 4 | 4 |
| **178** | 4 | 4 | 4 |
| **179** | 4 | 4 | 4 |
| **180** | 4 | 4 | 4 |
| **181** | 4 | 4 | 4 |
| **182** | 4 | 4 | 4 |
| **183** | 4 | 4 | 4 |
| **184** | 4 | 4 | 4 |
| **185** | 4 | 4 | 4 |
| **186** | 4 | 4 | 4 |
| **187** | 4 | 4 | 4 |
| **188** | 4 | 4 | 4 |
| **189** | 4 | 4 | 4 |
| **190** | 4 | 4 | 4 |
| **191** | 4 | 4 | 4 |
| **192** | 4 | 4 | 4 |
| **193** | 4 | 4 | 4 |
| **194** | 4 | 4 | 4 |
| **195** | 4 | 4 | 4 |
| **196** | 4 | 4 | 4 |

给出混淆矩阵，首先是没有降维之前的混淆矩阵。



降维后混淆矩阵为：

发现降维后，多预测正确一个第2类的事物。

## 代码

1. **from** scipy.io **import** loadmat
2. **from** sklearn **import** svm
3. **import** pandas as pd
4. **import** numpy as np
5. **from** sklearn.metrics **import** accuracy\_score
6. **from** sklearn.model\_selection **import** LeaveOneOut
7. **from** sklearn.decomposition **import** PCA
8. **import** matplotlib.pyplot as plt
9. **import** matplotlib as mpl
10. **from** sklearn.metrics **import** confusion\_matrix
11. **from** sklearn.metrics **import** recall\_score
12. **import** matplotlib.pyplot as plt
13. **def** confusion\_plt(guess,fact):
14. classes = list(set(fact))
15. classes.sort()
16. confusion = confusion\_matrix(guess, fact)
17. plt.imshow(confusion, cmap=plt.cm.Blues)
18. indices = range(len(confusion))
19. plt.xticks(indices, classes)
20. plt.yticks(indices, classes)
21. plt.colorbar()
22. plt.xlabel('guess')
23. plt.ylabel('fact')
24. **for** first\_index **in** range(len(confusion)):
25. **for** second\_index **in** range(len(confusion[first\_index])):
26. plt.text(first\_index, second\_index, confusion[first\_index][second\_index])
27. plt.show()
28. #读入数据
29. m = loadmat("data.mat")
30. data = pd.DataFrame(m["data"])
31. label = pd.DataFrame(m["label"])
32. #预测结果存入label\_hat1
33. label\_hat1 = pd.DataFrame(m["label"]).copy()
34. #留一法训练
35. loo=LeaveOneOut()
36. loo.get\_n\_splits(data)
37. **for** train\_index,test\_index **in** loo.split(data,label):
38. x\_train,x\_test=data.loc[train\_index],data.loc[test\_index]
39. y\_train,y\_test=label.loc[train\_index],label.loc[test\_index]
40. #clf = svm.SVC(C=1, kernel='linear', gamma='auto', decision\_function\_shape='ovr')         #高斯核测试
41. clf = svm.SVC(C=1, kernel='linear', decision\_function\_shape='ovr')
42. clf.fit(x\_train, y\_train.values.ravel())
43. label\_hat1.loc[test\_index] = clf.predict(x\_test)
44. #计算参考基值
45. count\_baseline = accuracy\_score(label,label\_hat1,False)
46. accuracy\_baseline = accuracy\_score(label,label\_hat1,True)
47. **print**("全部特征进行分类预测成功数量为"+str(count\_baseline)+"预测成功率为"+str(accuracy\_baseline))
48. confusion\_plt(label\_hat1.values.ravel(),label.values.ravel())
50. #主成分分析进行特征选择
51. data\_t = PCA(n\_components=170).fit\_transform(data)
52. #预测结果存入label\_hat2
53. label\_hat2 = pd.DataFrame(m["label"]).copy()
54. #留一法训练
55. loo=LeaveOneOut()
56. loo.get\_n\_splits(data\_t)
57. **for** train\_index,test\_index **in** loo.split(data\_t,label):
58. x\_train,x\_test=data\_t[train\_index],data\_t[test\_index]
59. y\_train,y\_test=label.loc[train\_index],label.loc[test\_index]
60. clf = svm.SVC(C=1, kernel='linear', decision\_function\_shape='ovr')
61. clf.fit(x\_train, y\_train.values.ravel())
62. label\_hat2.loc[test\_index] = clf.predict(x\_test)
63. #计算值
64. count\_new = accuracy\_score(label,label\_hat2,False)
65. accuracy\_new = accuracy\_score(label,label\_hat2,True)
66. **print**("降维后分类预测成功数量为"+str(count\_new)+"预测成功率为"+str(accuracy\_new))
67. confusion\_plt(label\_hat2.values.ravel(),label.values.ravel())
68. label\_new = pd.concat([label,label\_hat1,label\_hat2],axis=1)
69. label\_new.to\_excel('对比.xls')