Boxplots of 100 Training Error Rates and Test Error Rates for 6 methods

Lily (Lizheng) Zhou

November 16, 2019

1 Introduction

In this project, I repeat the following 100 times:

Randomly split the data in half, that is train and test.

Fitted the models with 6 methods:

1. LDA
2. QDA
3. KNN and tune K using 10-fold CV
4. LASSO logistic and tune lambda using 10-fold CV
5. Ridge logistic and tune lambda using 10 fold CV
6. Random Forest with m = sqrt(p) and 300 bootstrapped trees

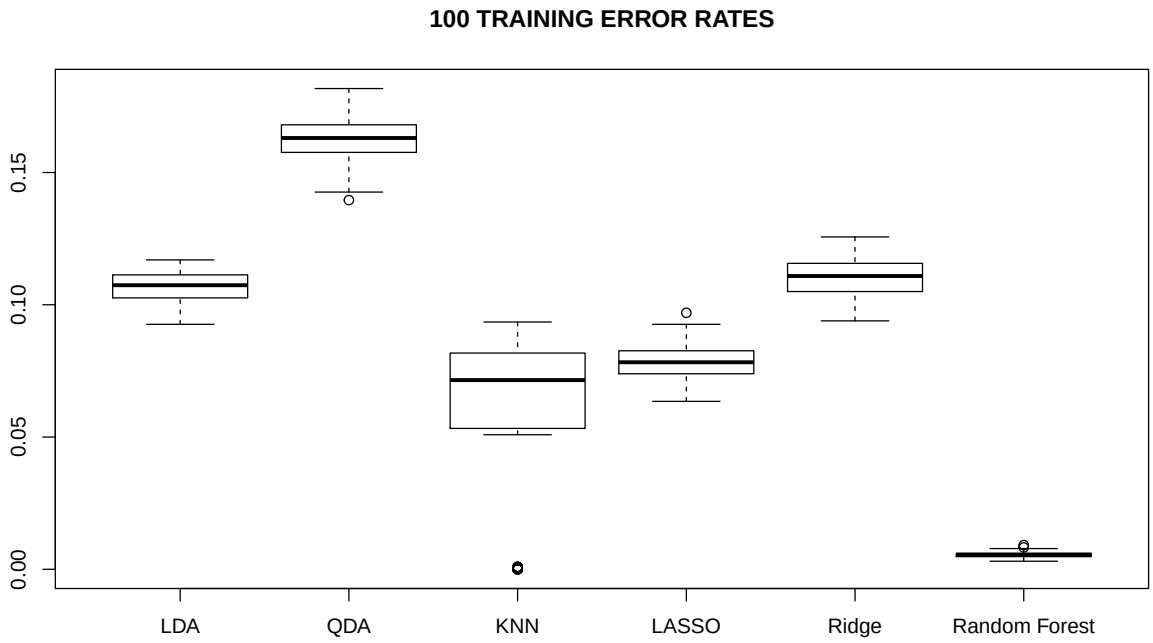
Later use the models to predict the responses for the TRAIN data and TEST data, and calculate the TRAINING error rate and TEST error rate.

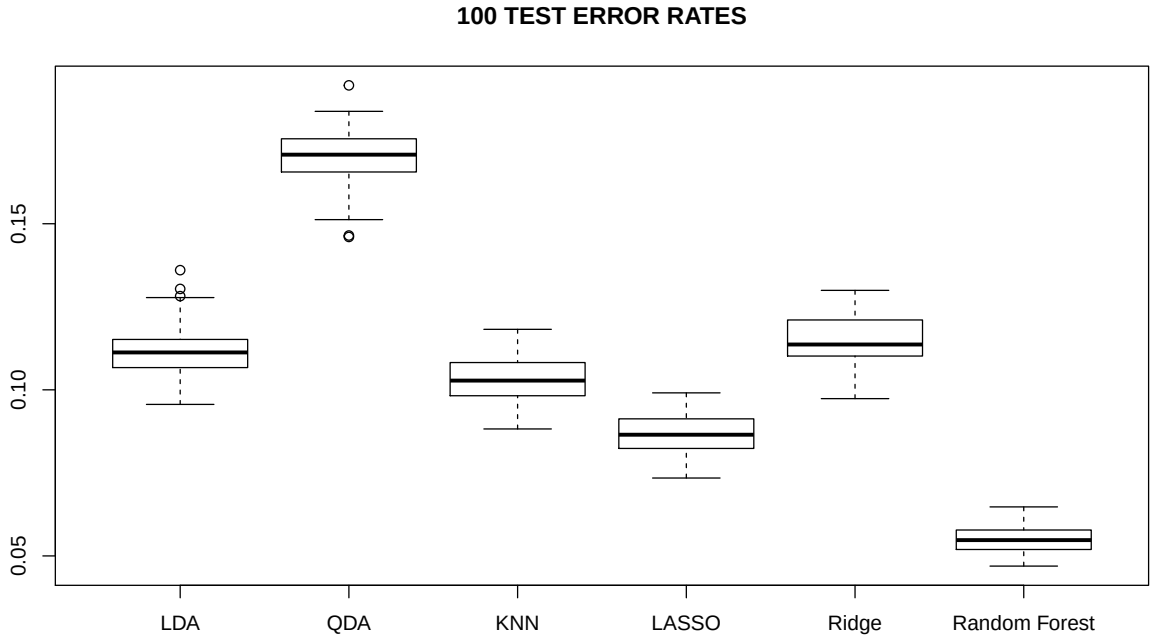
At last, I created side-by-side boxplots of the 100 Training Error Rates and Test Error Rates for the methods mentioned above.

Data Source: Spam Dataset https://web.stanford.edu/~hastie/ElemStatLearn/

2 Boxplots

Side-by-side boxplots of the 100 Training Error Rates and Test Error Rates for LDA, QDA with noise, KNN, Lasso, Ridge, Random Forest methods shown here.





3 Conclusion

Within one method, we see that a model's training error rate is lower than test error rate, test error rate is true error rate, while the training error rate is often overly optimistic.

From the Boxplots, we see that **Random Forest seems works best** for this dataset, test error rate and training error rate are all very low. While **QDA perform worst**, actually this is already adding noise to QDA, the error rate is still highest. Other methods perform between QDA and Random Forest.

Lasso and ridge are similar methods, while Lasso perform better in this dataset. **LDA** and **KNN** perform ok. Also since I used Train data and `tune.knn()` function to choose optimal k, k value finally use are different (most k are from 1-10), so the variance of training error rate of KNN is larger than others.

4 Appendix

4.1 Matrix of Training Error Rates and Test Error Rates

Training Error Rate Matrix:

```
##               LDA       QDA          KNN      LASSO      Ridge Random Forest
##   [1,] 0.10652174 0.1621739 0.0682608696 0.07173913 0.11304348   0.005217391
##   [2,] 0.11304348 0.1652174 0.0717391304 0.08130435 0.11217391   0.005217391
##   [3,] 0.10565217 0.1569565 0.0000000000 0.07521739 0.10913043   0.006086957
##   [4,] 0.10086957 0.1608696 0.0691304348 0.07130435 0.10304348   0.003913043
##   [5,] 0.11173913 0.1713043 0.0913043478 0.08043478 0.11086957   0.005652174
##   [6,] 0.10391304 0.1643478 0.0839130435 0.08260870 0.10869565   0.006521739
##   [7,] 0.09913043 0.1626087 0.0730434783 0.07043478 0.09608696   0.005217391
##   [8,] 0.11521739 0.1630435 0.0826086957 0.07695652 0.11826087   0.009130435
##   [9,] 0.11130435 0.1608696 0.0534782609 0.07434783 0.11434783   0.004347826
##  [10,] 0.11608696 0.1665217 0.0856521739 0.08478261 0.11347826   0.004782609
##  [11,] 0.10173913 0.1617391 0.0765217391 0.08086957 0.10347826   0.005217391
##  [12,] 0.10739130 0.1617391 0.0865217391 0.08521739 0.12173913   0.006086957
##  [13,] 0.11565217 0.1647826 0.0008695652 0.07826087 0.11086957   0.005217391
##  [14,] 0.09695652 0.1495652 0.0556521739 0.07000000 0.10043478   0.005217391
##  [15,] 0.10956522 0.1617391 0.0700000000 0.08652174 0.11347826   0.005217391
##  [16,] 0.11043478 0.1691304 0.0008695652 0.08260870 0.11347826   0.004782609
##  [17,] 0.09391304 0.1604348 0.0752173913 0.06739130 0.09695652   0.006086957
##  [18,] 0.11608696 0.1760870 0.0821739130 0.08391304 0.12043478   0.004782609
##  [19,] 0.10217391 0.1591304 0.0556521739 0.08043478 0.10217391   0.004347826
##  [20,] 0.09826087 0.1600000 0.0004347826 0.07086957 0.10304348   0.005652174
##  [21,] 0.09956522 0.1660870 0.0800000000 0.07521739 0.10782609   0.004782609
##  [22,] 0.11000000 0.1560870 0.0856521739 0.07043478 0.10434783   0.004347826
##  [23,] 0.10304348 0.1495652 0.0852173913 0.07260870 0.10739130   0.004782609
##  [24,] 0.10478261 0.1426087 0.0004347826 0.07434783 0.11521739   0.004782609
##  [25,] 0.10739130 0.1552174 0.0665217391 0.07913043 0.11478261   0.005217391
##  [26,] 0.10913043 0.1756522 0.0513043478 0.08521739 0.11565217   0.005652174
##  [27,] 0.10869565 0.1730435 0.0000000000 0.08130435 0.11130435   0.003913043
##  [28,] 0.09782609 0.1573913 0.0000000000 0.07043478 0.09652174   0.005217391
##  [29,] 0.11434783 0.1769565 0.0826086957 0.08391304 0.12347826   0.007826087
##  [30,] 0.10956522 0.1452174 0.0860869565 0.06695652 0.11086957   0.006086957
##  [31,] 0.11043478 0.1717391 0.0913043478 0.08086957 0.12260870   0.006521739
##  [32,] 0.11173913 0.1682609 0.0556521739 0.07782609 0.11521739   0.006086957
##  [33,] 0.10260870 0.1630435 0.0769565217 0.08782609 0.10913043   0.005217391
##  [34,] 0.10478261 0.1634783 0.0826086957 0.08173913 0.10739130   0.005652174
##  [35,] 0.10043478 0.1652174 0.0804347826 0.07826087 0.11347826   0.007391304
##  [36,] 0.11347826 0.1652174 0.0004347826 0.07608696 0.11043478   0.005652174
##  [37,] 0.10956522 0.1652174 0.0543478261 0.08043478 0.11695652   0.004347826
##  [38,] 0.10478261 0.1395652 0.0900000000 0.07304348 0.10739130   0.005652174
```

```
##  [39,] 0.10826087 0.1473913 0.0773913043 0.07565217 0.10521739   0.005652174
##  [40,] 0.10086957 0.1513043 0.0821739130 0.06956522 0.10652174   0.006521739
##  [41,] 0.10521739 0.1600000 0.0660869565 0.07739130 0.10217391   0.007391304
##  [42,] 0.10826087 0.1691304 0.0008695652 0.08173913 0.11434783   0.006521739
##  [43,] 0.10521739 0.1586957 0.0000000000 0.07826087 0.11913043   0.003913043
##  [44,] 0.10260870 0.1678261 0.0008695652 0.07391304 0.11086957   0.006956522
##  [45,] 0.11478261 0.1717391 0.0817391304 0.09260870 0.11739130   0.005217391
##  [46,] 0.10739130 0.1604348 0.0582608696 0.07695652 0.10391304   0.005652174
##  [47,] 0.10260870 0.1678261 0.0886956522 0.08173913 0.11347826   0.003913043
##  [48,] 0.11000000 0.1530435 0.0778260870 0.08304348 0.11565217   0.005652174
##  [49,] 0.10391304 0.1491304 0.0530434783 0.07695652 0.12130435   0.006521739
##  [50,] 0.09260870 0.1773913 0.0665217391 0.07434783 0.10173913   0.004782609
##  [51,] 0.10565217 0.1617391 0.0008695652 0.07217391 0.10347826   0.005652174
##  [52,] 0.11652174 0.1695652 0.0752173913 0.08478261 0.11608696   0.007391304
##  [53,] 0.10043478 0.1552174 0.0000000000 0.07565217 0.10391304   0.003478261
##  [54,] 0.10347826 0.1765217 0.0004347826 0.07608696 0.11086957   0.006521739
##  [55,] 0.11217391 0.1682609 0.0004347826 0.08826087 0.11913043   0.006521739
##  [56,] 0.10434783 0.1556522 0.0900000000 0.08391304 0.10956522   0.004347826
##  [57,] 0.11217391 0.1656522 0.0730434783 0.08739130 0.12304348   0.003913043
##  [58,] 0.10130435 0.1491304 0.0826086957 0.07608696 0.10565217   0.005652174
##  [59,] 0.10260870 0.1617391 0.0926086957 0.07652174 0.10347826   0.006956522
##  [60,] 0.10434783 0.1508696 0.0660869565 0.07086957 0.10608696   0.004782609
##  [61,] 0.09782609 0.1582609 0.0569565217 0.08043478 0.09434783   0.004347826
##  [62,] 0.09695652 0.1569565 0.0791304348 0.06782609 0.10173913   0.006086957
##  [63,] 0.10869565 0.1808696 0.0700000000 0.08217391 0.11217391   0.006086957
##  [64,] 0.11565217 0.1630435 0.0734782609 0.08391304 0.12217391   0.003913043
##  [65,] 0.09304348 0.1556522 0.0704347826 0.06826087 0.09565217   0.004347826
##  [66,] 0.09652174 0.1634783 0.0000000000 0.07391304 0.10652174   0.004782609
##  [67,] 0.10739130 0.1647826 0.0573913043 0.08434783 0.11260870   0.006521739
##  [68,] 0.10304348 0.1604348 0.0000000000 0.07173913 0.10434783   0.003478261
##  [69,] 0.10956522 0.1547826 0.0734782609 0.08130435 0.11652174   0.004782609
##  [70,] 0.11173913 0.1686957 0.0769565217 0.08521739 0.10826087   0.003913043
##  [71,] 0.11043478 0.1552174 0.0843478261 0.08000000 0.11391304   0.007391304
##  [72,] 0.11391304 0.1660870 0.0843478261 0.07913043 0.11652174   0.003913043
##  [73,] 0.10434783 0.1517391 0.0547826087 0.07173913 0.10391304   0.005217391
##  [74,] 0.10434783 0.1608696 0.0778260870 0.07782609 0.11739130   0.006086957
##  [75,] 0.11347826 0.1782609 0.0508695652 0.08826087 0.12000000   0.005652174
##  [76,] 0.11130435 0.1552174 0.0643478261 0.07869565 0.12565217   0.005217391
##  [77,] 0.09478261 0.1578261 0.0817391304 0.06347826 0.09608696   0.004782609
##  [78,] 0.10086957 0.1660870 0.0752173913 0.06869565 0.10608696   0.006956522
##  [79,] 0.10956522 0.1717391 0.0004347826 0.08000000 0.11173913   0.006086957
##  [80,] 0.10782609 0.1669565 0.0000000000 0.08608696 0.11391304   0.006086957
##  [81,] 0.11086957 0.1704348 0.0726086957 0.07565217 0.10608696   0.007826087
##  [82,] 0.10391304 0.1465217 0.0660869565 0.07173913 0.10521739   0.004782609
##  [83,] 0.11086957 0.1617391 0.0795652174 0.07739130 0.11391304   0.005652174
##  [84,] 0.10652174 0.1617391 0.0713043478 0.07869565 0.10304348   0.004347826
##  [85,] 0.11000000 0.1743478 0.0591304348 0.07260870 0.11086957   0.006086957
##  [86,] 0.11217391 0.1669565 0.0004347826 0.08565217 0.11260870   0.005217391
##  [87,] 0.11000000 0.1600000 0.0521739130 0.08173913 0.11826087   0.006956522
##  [88,] 0.10869565 0.1665217 0.0678260870 0.07826087 0.10782609   0.003913043
##  [89,] 0.11521739 0.1678261 0.0782608696 0.09695652 0.12173913   0.005652174
##  [90,] 0.10043478 0.1817391 0.0921739130 0.08043478 0.10391304   0.006956522
##  [91,] 0.11652174 0.1647826 0.0843478261 0.09086957 0.12130435   0.005652174
##  [92,] 0.11043478 0.1639130 0.0678260870 0.07521739 0.11043478   0.004782609
```

```
##  [93,] 0.11391304 0.1608696 0.0730434783 0.08000000 0.10826087   0.003043478
##  [94,] 0.11000000 0.1704348 0.0782608696 0.07826087 0.11217391   0.004347826
##  [95,] 0.09782609 0.1660870 0.0773913043 0.08043478 0.10478261   0.006086957
##  [96,] 0.09956522 0.1460870 0.0682608696 0.07260870 0.09391304   0.003478261
##  [97,] 0.11434783 0.1673913 0.0934782609 0.08260870 0.11565217   0.008260870
##  [98,] 0.10695652 0.1686957 0.0830434783 0.07217391 0.10739130   0.003913043
##  [99,] 0.11434783 0.1700000 0.0808695652 0.08391304 0.11739130   0.005652174
## [100,] 0.11695652 0.1782609 0.0004347826 0.08260870 0.12304348   0.007826087
```

Test Error Rate Matrix:

```
##               LDA       QDA        KNN      LASSO      Ridge Random Forest
##   [1,] 0.09778357 0.1616688 0.09821817 0.08344198 0.09908735    0.05432421
##   [2,] 0.10908301 0.1642764 0.09995654 0.08387658 0.11082138    0.05649718
##   [3,] 0.12168622 0.1755758 0.10908301 0.08952629 0.12690135    0.05736636
##   [4,] 0.11386354 0.1751412 0.10039113 0.09343764 0.11169057    0.05432421
##   [5,] 0.13037810 0.1547153 0.11386354 0.09387223 0.12733594    0.05388961
##   [6,] 0.11212516 0.1581921 0.10777923 0.09039548 0.11299435    0.05953933
##   [7,] 0.10647545 0.1720991 0.09734898 0.07779226 0.10995219    0.05345502
##   [8,] 0.10908301 0.1677532 0.10039113 0.09169926 0.11255976    0.06040852
##   [9,] 0.12646675 0.1694915 0.10560626 0.08996089 0.12429379    0.05736636
##  [10,] 0.10734463 0.1673186 0.10256410 0.08822251 0.10691004    0.04997827
##  [11,] 0.10212951 0.1655802 0.10343329 0.08474576 0.10473707    0.06214689
##  [12,] 0.10126032 0.1773142 0.09734898 0.07866145 0.09952195    0.05258583
##  [13,] 0.11169057 0.1464581 0.11212516 0.08604954 0.11473272    0.05780096
##  [14,] 0.12820513 0.1742721 0.10647545 0.09083007 0.12255541    0.06388527
##  [15,] 0.10430248 0.1738375 0.09213385 0.07953064 0.10691004    0.05388961
##  [16,] 0.10777923 0.1629726 0.09734898 0.08604954 0.10691004    0.05215124
##  [17,] 0.12125163 0.1825293 0.10386788 0.09691439 0.12168622    0.05649718
##  [18,] 0.11690569 0.1651456 0.10082573 0.09213385 0.12081704    0.04737071
##  [19,] 0.12125163 0.1673186 0.10299870 0.09865276 0.12559757    0.05910474
##  [20,] 0.10473707 0.1621034 0.11386354 0.08126901 0.11994785    0.05736636
##  [21,] 0.10604085 0.1755758 0.09865276 0.08170361 0.11038679    0.06214689
##  [22,] 0.10647545 0.1773142 0.10691004 0.08300739 0.11082138    0.05953933
##  [23,] 0.10951760 0.1586267 0.10821382 0.07648848 0.11212516    0.05302043
##  [24,] 0.11907866 0.1751412 0.11429813 0.09213385 0.12299000    0.06475445
##  [25,] 0.11342894 0.1586267 0.09517601 0.09474142 0.12516297    0.05258583
##  [26,] 0.12777053 0.1755758 0.11038679 0.09213385 0.12603216    0.05432421
##  [27,] 0.11864407 0.1707953 0.10821382 0.08474576 0.11169057    0.05258583
##  [28,] 0.10777923 0.1707953 0.11212516 0.08474576 0.10691004    0.05823555
##  [29,] 0.10821382 0.1690569 0.10299870 0.08126901 0.11994785    0.05432421
##  [30,] 0.12299000 0.1760104 0.11820947 0.09821817 0.12472838    0.05649718
##  [31,] 0.09647979 0.1616688 0.09517601 0.07344633 0.10430248    0.04823990
##  [32,] 0.11864407 0.1707953 0.10082573 0.09865276 0.12429379    0.05432421
##  [33,] 0.11342894 0.1712299 0.10647545 0.08996089 0.11386354    0.05084746
##  [34,] 0.10734463 0.1794872 0.10560626 0.09083007 0.11082138    0.05171664
##  [35,] 0.10691004 0.1742721 0.09995654 0.08648414 0.11082138    0.05388961
##  [36,] 0.10430248 0.1673186 0.10343329 0.07909605 0.11386354    0.05475880
##  [37,] 0.10691004 0.1664494 0.11212516 0.07909605 0.11169057    0.04823990
##  [38,] 0.11125598 0.1603651 0.11255976 0.08213820 0.11255976    0.05084746
##  [39,] 0.11125598 0.1738375 0.09908735 0.09734898 0.12125163    0.05910474
##  [40,] 0.10951760 0.1807910 0.11473272 0.07953064 0.11169057    0.05606258
##  [41,] 0.11516732 0.1773142 0.09517601 0.09083007 0.11516732    0.05606258
##  [42,] 0.12125163 0.1703607 0.09604520 0.08648414 0.12168622    0.05606258
```

```
##  [43,] 0.10212951 0.1577575 0.09995654 0.07909605 0.11734029    0.05693177
##  [44,] 0.10821382 0.1616688 0.10560626 0.08604954 0.11386354    0.05649718
##  [45,] 0.09561060 0.1760104 0.08909170 0.07431551 0.09821817    0.04737071
##  [46,] 0.11994785 0.1677532 0.11038679 0.09387223 0.11994785    0.06388527
##  [47,] 0.11864407 0.1616688 0.11342894 0.09561060 0.12212082    0.05693177
##  [48,] 0.10604085 0.1716645 0.10777923 0.08257279 0.11038679    0.05084746
##  [49,] 0.10864841 0.1690569 0.09778357 0.08083442 0.12559757    0.05171664
##  [50,] 0.10169492 0.1651456 0.10169492 0.08387658 0.10517166    0.05867014
##  [51,] 0.10995219 0.1686223 0.11212516 0.07996523 0.10908301    0.05128205
##  [52,] 0.11299435 0.1707953 0.10343329 0.08604954 0.10777923    0.05084746
##  [53,] 0.09995654 0.1734029 0.11429813 0.08691873 0.11038679    0.05475880
##  [54,] 0.10734463 0.1768796 0.10647545 0.08735332 0.11386354    0.05475880
##  [55,] 0.11342894 0.1690569 0.10995219 0.09169926 0.11560191    0.06214689
##  [56,] 0.11777488 0.1677532 0.10777923 0.08518036 0.12125163    0.05432421
##  [57,] 0.11734029 0.1512386 0.10039113 0.09908735 0.12950891    0.05823555
##  [58,] 0.10951760 0.1690569 0.11299435 0.08952629 0.11864407    0.06345067
##  [59,] 0.10517166 0.1760104 0.09952195 0.07909605 0.10647545    0.04693611
##  [60,] 0.11516732 0.1812256 0.10212951 0.09083007 0.11907866    0.05780096
##  [61,] 0.11429813 0.1777488 0.10604085 0.09430682 0.11777488    0.05171664
##  [62,] 0.09821817 0.1712299 0.09691439 0.07779226 0.09734898    0.05780096
##  [63,] 0.11169057 0.1720991 0.09474142 0.09213385 0.12168622    0.05258583
##  [64,] 0.11125598 0.1664494 0.09604520 0.08822251 0.12081704    0.05084746
##  [65,] 0.11212516 0.1677532 0.10951760 0.08170361 0.11038679    0.05693177
##  [66,] 0.10691004 0.1707953 0.11777488 0.08300739 0.10951760    0.05606258
##  [67,] 0.11994785 0.1755758 0.10343329 0.09430682 0.12168622    0.05780096
##  [68,] 0.11342894 0.1655802 0.11429813 0.08996089 0.11125598    0.06214689
##  [69,] 0.09908735 0.1799218 0.08865711 0.08170361 0.10864841    0.05388961
##  [70,] 0.11647110 0.1807910 0.10169492 0.08952629 0.11212516    0.04954368
##  [71,] 0.10126032 0.1651456 0.10082573 0.08778792 0.11125598    0.05606258
##  [72,] 0.12429379 0.1664494 0.10430248 0.09039548 0.12516297    0.05432421
##  [73,] 0.11603651 0.1742721 0.11082138 0.09604520 0.11820947    0.05345502
##  [74,] 0.11038679 0.1777488 0.10212951 0.08257279 0.11212516    0.06084311
##  [75,] 0.11473272 0.1734029 0.10647545 0.09604520 0.12299000    0.04693611
##  [76,] 0.11429813 0.1460235 0.10039113 0.08648414 0.12994350    0.05953933
##  [77,] 0.10386788 0.1720991 0.11820947 0.08431117 0.10560626    0.05867014
##  [78,] 0.10908301 0.1655802 0.09604520 0.07779226 0.10777923    0.05475880
##  [79,] 0.10647545 0.1564537 0.09647979 0.08648414 0.11212516    0.05171664
##  [80,] 0.11473272 0.1734029 0.11516732 0.08387658 0.11647110    0.05171664
##  [81,] 0.11299435 0.1668840 0.08822251 0.09083007 0.11342894    0.05823555
##  [82,] 0.10734463 0.1594959 0.09821817 0.08518036 0.11473272    0.05867014
##  [83,] 0.10386788 0.1755758 0.09343764 0.07953064 0.10517166    0.05128205
##  [84,] 0.11125598 0.1816601 0.09995654 0.09083007 0.10777923    0.04954368
##  [85,] 0.11299435 0.1738375 0.10082573 0.08909170 0.11690569    0.05997392
##  [86,] 0.11082138 0.1720991 0.09474142 0.08344198 0.11820947    0.04780530
##  [87,] 0.11603651 0.1816601 0.10647545 0.09213385 0.12516297    0.05258583
##  [88,] 0.12125163 0.1738375 0.10169492 0.09734898 0.12255541    0.05388961
##  [89,] 0.11255976 0.1534116 0.09083007 0.09213385 0.11777488    0.05258583
##  [90,] 0.09821817 0.1664494 0.09995654 0.08778792 0.10691004    0.05693177
##  [91,] 0.11082138 0.1760104 0.08996089 0.07953064 0.10951760    0.05302043
##  [92,] 0.11255976 0.1703607 0.09647979 0.08778792 0.10691004    0.05171664
##  [93,] 0.11169057 0.1694915 0.10256410 0.08778792 0.10430248    0.05519339
##  [94,] 0.11125598 0.1712299 0.09734898 0.08518036 0.11473272    0.05953933
##  [95,] 0.10647545 0.1916558 0.10560626 0.09039548 0.11125598    0.05475880
##  [96,] 0.13602781 0.1734029 0.10386788 0.09691439 0.12299000    0.06431986
```

```
##  [97,] 0.11038679 0.1534116 0.10082573 0.08213820 0.11690569    0.04823990
##  [98,] 0.11342894 0.1803564 0.11255976 0.07909605 0.11082138    0.05215124
##  [99,] 0.11255976 0.1681877 0.10647545 0.08474576 0.11603651    0.04954368
## [100,] 0.10647545 0.1838331 0.09300304 0.08648414 0.11255976    0.05519339
```

Matrix Summary:

Training Error Rate Matrix Summary:

```
##       LDA               QDA               KNN               LASSO
##  Min.   :0.09261   Min.   :0.1396   Min.   :0.00000   Min.   :0.06348
##  1st Qu.:0.10261   1st Qu.:0.1577   1st Qu.:0.05337   1st Qu.:0.07391
##  Median :0.10739   Median :0.1630   Median :0.07152   Median :0.07826
##  Mean   :0.10683   Mean   :0.1628   Mean   :0.05866   Mean   :0.07836
##  3rd Qu.:0.11130   3rd Qu.:0.1679   3rd Qu.:0.08174   3rd Qu.:0.08261
##  Max.   :0.11696   Max.   :0.1817   Max.   :0.09348   Max.   :0.09696
##      Ridge         Random Forest
##  Min.   :0.09391   Min.   :0.003043
##  1st Qu.:0.10511   1st Qu.:0.004783
##  Median :0.11087   Median :0.005435
##  Mean   :0.11046   Mean   :0.005478
##  3rd Qu.:0.11565   3rd Qu.:0.006087
##  Max.   :0.12565   Max.   :0.009130
```

Test Error Rate Matrix Summary:

```
##       LDA               QDA               KNN               LASSO
##  Min.   :0.09561   Min.   :0.1460   Min.   :0.08822   Min.   :0.07345
##  1st Qu.:0.10680   1st Qu.:0.1656   1st Qu.:0.09822   1st Qu.:0.08246
##  Median :0.11126   Median :0.1708   Median :0.10278   Median :0.08648
##  Mean   :0.11143   Mean   :0.1698   Mean   :0.10340   Mean   :0.08705
##  3rd Qu.:0.11517   3rd Qu.:0.1756   3rd Qu.:0.10821   3rd Qu.:0.09105
##  Max.   :0.13603   Max.   :0.1917   Max.   :0.11821   Max.   :0.09909
##      Ridge         Random Forest
##  Min.   :0.09735   Min.   :0.04694
##  1st Qu.:0.11028   1st Qu.:0.05204
##  Median :0.11365   Median :0.05476
##  Mean   :0.11461   Mean   :0.05508
##  3rd Qu.:0.12093   3rd Qu.:0.05780
##  Max.   :0.12994   Max.   :0.06475
```

4.2 R Code

```
knitr::opts_chunk$set(echo = TRUE)
set.seed(1)
# load library
library(MASS)
library(class)
library(glmnet )
library(tree)
library(e1071)
```

```r
library(randomForest)

# Exploratory Data Analysis
spam = read.table("D:/d/Courses/STA/STA 9891/HW/HW4/spam.data.txt")

# Matrix create
train_err.matrix = matrix(0, nrow = 100, ncol = 6)
colnames(train_err.matrix) = c("LDA","QDA","KNN","LASSO","Ridge","Random Forest")

test_err.matrix = matrix(0, nrow = 100, ncol = 6)
colnames(test_err.matrix) = c("LDA","QDA","KNN","LASSO","Ridge","Random Forest")

# Standardize the data
n = nrow(spam)
p = ncol(spam) - 1
spam[ ,1:p] = data.frame(scale(spam[ ,1:p]))

# add noise for QDA
noise = matrix(rnorm(n=131100, mean=0, sd=0.01), nrow=2300, ncol=57)

# for loop
for (i in 1:100) {
  # Split in half
  train = sample(n, n/2)
  spam.train = spam[train, ]
  spam.test = spam[-train, ]
  V58.train = spam[,58][train]
  V58.test = spam[,58][-train]
  x.train = spam.train[,-58]
  x.test = spam.test[,-58]

  # LDA
  fit.lda = lda(V58 ~ ., data = spam, subset = train)
  # train
  lda.pred.train = predict(fit.lda, spam.train)
  lda.class.train=lda.pred.train$class
  train_err.matrix[i,1] = mean(lda.class.train != V58.train)
  # test
  lda.pred.test = predict(fit.lda, spam.test)
  lda.class.test=lda.pred.test$class
  test_err.matrix[i,1] = mean(lda.class.test != V58.test)

  # QDA with noise
  spam.train.noise = data.frame(data.matrix(spam.train[,-58])+noise, 'V58'=spam.train[,58])
  fit.qda = qda(V58 ~ ., data = spam.train.noise)
  # train
  qda.pred.train = predict(fit.qda, spam.train)
  qda.class.train=qda.pred.train$class
  train_err.matrix[i,2] = mean(qda.class.train != V58.train)
  # test
  qda.pred.test = predict(fit.qda, spam.test)
  qda.class.test=qda.pred.test$class
  test_err.matrix[i,2] = mean(qda.class.test != V58.test)
```

```
# KNN
# tune K using 10-fold CV for KNN
optimal_k = tune.knn(as.matrix(x.train), as.factor(V58.train), k=1:20,
                     tunecontrol=tune.control(sampling="cross"), cross=10)
best_k = optimal_k$best.parameters$k
# train
knn.pred.train = knn(x.train, x.train, V58.train, k=best_k)
train_err.matrix[i,3] = mean(knn.pred.train != V58.train)
# test
knn.pred.test = knn(x.train, x.test, V58.train, k=best_k)
test_err.matrix[i,3] = mean(knn.pred.test != V58.test)

# LASSO and Ridge
x = model.matrix(V58 ~ ., spam)[,-1]
y = spam$V58

# Lasso
# Find the optimal lambda value via cross validation
cv.out.lasso = cv.glmnet(x[train,], y[train], alpha=1, family="binomial",
                         intercept=TRUE, standardize=FALSE, type.measure="class")
bestlam.lasso = cv.out.lasso$lambda.min
# Fit a lasso regression model
lasso.fit = glmnet(x[train,],y[train], alpha=1, lambda=bestlam.lasso, family="binomial",
                   intercept=TRUE, standardize=FALSE)
# Compute the train error
lasso.prob.train = predict(lasso.fit, s=bestlam.lasso, newx=x[train,])
lasso.pred.train = V58.train
lasso.pred.train[lasso.prob.train>0.5] = 1
lasso.pred.train[lasso.prob.train<0.5] = 0
train_err.matrix[i,4] = mean(y[train] != lasso.pred.train)
# Compute the test error
lasso.prob.test = predict(lasso.fit, s=bestlam.lasso, newx=x[-train,])
lasso.pred.test = V58.test
lasso.pred.test[lasso.prob.test>0.5] = 1
lasso.pred.test[lasso.prob.test<0.5] = 0
test_err.matrix[i,4] = mean(y[-train] != lasso.pred.test)

# Ridge
# Find the optimal lambda value via cross validation
cv.out.ridge = cv.glmnet(x[train,], y[train], alpha=0, family="binomial",
                         intercept=TRUE, standardize=FALSE, type.measure="class")
bestlam.ridge = cv.out.ridge$lambda.min
# Fit a ridge regression model
ridge.fit = glmnet(x[train,],y[train], alpha=0, lambda=bestlam.ridge, family="binomial",
                   intercept=TRUE, standardize=FALSE)
# Compute the train error
ridge.prob.train = predict(ridge.fit, s=bestlam.ridge, newx=x[train,])
ridge.pred.train = V58.train
ridge.pred.train[ridge.prob.train>0.5] = 1
ridge.pred.train[ridge.prob.train<0.5] = 0
train_err.matrix[i,5] = mean(y[train] != ridge.pred.train)
# Compute the test error
ridge.prob.test = predict(ridge.fit, s=bestlam.ridge, newx=x[-train,])
```

```
  ridge.pred.test = V58.test
  ridge.pred.test[ridge.prob.test>0.5] = 1
  ridge.pred.test[ridge.prob.test<0.5] = 0
  test_err.matrix[i,5] = mean(y[-train] != ridge.pred.test)

  # Random Forest
  fit.rf = randomForest(V58 ~ ., data=spam, subset=train,
                        mtry=sqrt(p), ntree=300, importance=TRUE)
  # train
  rf.prod.train = predict(fit.rf, newdata=spam.train)
  rf.pred.train = V58.train
  rf.pred.train[rf.prod.train>0.5] = 1
  rf.pred.train[rf.prod.train<0.5] = 0
  train_err.matrix[i,6] = mean(V58.train != rf.pred.train)

  # test
  rf.prod.test = predict(fit.rf, newdata=spam.test)
  rf.pred.test = V58.test
  rf.pred.test[rf.prod.test>0.5] = 1
  rf.pred.test[rf.prod.test<0.5] = 0
  test_err.matrix[i,6] = mean(V58.test != rf.pred.test)
}


train_err.matrix
test_err.matrix

# boxplots
par(mfrow=c(2,1))
boxplot(train_err.matrix, main = "100 TRAINING ERROR RATES")
boxplot(test_err.matrix, main = "100 TEST ERROR RATES")
# Training Error Rate Matrix
train_err.matrix
# Test Error Rate Matrix
test_err.matrix
# Training Error Rate Matrix Summary
summary(train_err.matrix)
# Test Error Rate Matrix Summary
summary(test_err.matrix)
```