Final Project

ST557

Nick Sun

December 7, 2019

Part 1

Let's compare red and white wines!

Part a.

Let's first compare the mean vectors between the red and white wines. we can do this with some variant of a good old Hotelling's two sample T^2 test, after checking the determinants of the covariance matrix for the respective datasets,

```
## [1] 3.478418e-11
```

```
## [1] 1.701408e-11
```

The determinants aren't too different, so we should be fine with using the pooled covariance Hotelling's.

Here are the sample mean vectors for the 11 variables in each wine dataset:

```
## [1] "Red Wines"
```

Table 1: Table continues below

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides
8.32	0.5278	0.271	2.539	0.08747

free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
15.87	46.47	0.9967	3.311	0.6581	10.42

```
## [1] "White Wines"
```

Table 3: Table continues below

fixed acidity	volatile acidity	citric acid	residual sugar	chlorides
6.855	0.2782	0.3342	6.391	0.04577

free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
35.31	138.4	0.994	3.188	0.4898	10.51

Here are the results of our Hotelling's T^2 test!

```
##          [,1]
## [1,] 40427.92
```

```
## [1] 21.07965
```

Here is the unequal covariance assumption T^2 test as well for good measure:

```
##          [,1]
## [1,] 29432.28
```

```
## [1] 21.02607
```

We have **strong** evidence that the population vectors of red and white wines are very different!

Where are they the most different? We can do individual level t-tests for this:

```
##  [1] "fixed acidity"        "volatile acidity"     "citric acid"
##  [4] "residual sugar"       "chlorides"            "free sulfur dioxide"
##  [7] "total sulfur dioxide" "density"              "pH"
## [10] "sulphates"            "alcohol"              "quality"
```

Table 5: Table continues below

	estimate.mean of x	estimate.mean of y
t.fixedacidity	8.32	6.855
t.volatileacidity	0.5278	0.2782
t.citricacid	8.32	6.855
t.residualsugar	2.539	6.391
t.chlorides	0.08747	0.04577
t.freesulphurdioxide	15.87	35.31
t.totalsulfurdioxide	46.47	138.4
t.density	0.9967	0.994
t.pH	3.311	3.188
t.sulphates	0.6581	0.4898
t.alcohol	10.42	10.51

	statistic.t	p.value
t.fixedacidity	32.42	5.668e-183
t.volatileacidity	53.06	0
t.citricacid	32.42	5.668e-183
t.residualsugar	-47.8	0
t.chlorides	34.24	6.095e-199
t.freesulphurdioxide	-54.43	0

Table 8: APER for Wine Model

x
0.0053929

	statistic.t	p.value
t.totalsulfurdioxide	-89.87	0
t.density	42.71	0
t.pH	27.78	2.342e-149
t.sulphates	37.06	1.679e-231
t.alcohol	-2.859	0.004278

Looks like all the variables invidually are significantly different? Should double check this result.

Part b.

Now let's come up with a classification rule! What is a good rule that will separate the red wines and the white wines?

Let's create a test and training set first and then try a few different methods!

Let's try Linear Discriminant Analysis first.

	LD1
`fixed acidity`	0.318
`volatile acidity`	-3.066
`citric acid`	0.9661
`residual sugar`	0.3484
chlorides	-5.075
`free sulfur dioxide`	-0.01942
`total sulfur dioxide`	0.02025
density	-895.1
pH	0.9943
sulphates	-0.8897
alcohol	-0.8002

Calculating the APER for the LDA model.

Wow this APER is very low… amazing~~~~

Just for fun, let's also do QDA…

This error rate is also very low, but not as low as LDA surprisingly enough.

We can also try a CART model, which is nice since it can produce a nice visual flowchart to follow.

```
## Call:
## rpart(formula = wine ~ ., data = subset(training, select = -quality))
##   n= 5199
##
##           CP nsplit  rel error    xerror        xstd
## 1 0.70546875      0 1.00000000 1.0000000 0.024267371
## 2 0.06562500      1 0.29453125 0.2992188 0.014715423
```

```
## 3 0.06171875      2 0.22890625 0.2296875 0.013011378
## 4 0.02031250      4 0.10546875 0.1203125 0.009550389
## 5 0.01562500      5 0.08515625 0.1023438 0.008828442
## 6 0.01000000      6 0.06953125 0.0921875 0.008389686
##
## Variable importance
## total sulfur dioxide            chlorides  free sulfur dioxide
##                   34                   21                   13
##     volatile acidity        fixed acidity          citric acid
##                   11                    7                    7
##              density            sulphates       residual sugar
##                    3                    2                    2
##
## Node number 1: 5199 observations,    complexity param=0.7054688
##   predicted class=whites  expected loss=0.2462012  P(node) =1
##     class counts:  1280  3919
##    probabilities: 0.246 0.754
##   left son=2 (1129 obs) right son=3 (4070 obs)
##   Primary splits:
##       total sulfur dioxide < 67.5     to the left,  improve=1232.5940, (0 missing)
##       chlorides            < 0.0615   to the right, improve=1198.2530, (0 missing)
##       volatile acidity     < 0.4125   to the right, improve= 762.6678, (0 missing)
##       free sulfur dioxide  < 17.5     to the left,  improve= 488.3827, (0 missing)
##       sulphates            < 0.545    to the right, improve= 407.1102, (0 missing)
##   Surrogate splits:
##       free sulfur dioxide < 13.5     to the left,  agree=0.864, adj=0.376, (0 split)
##       chlorides           < 0.0635   to the right, agree=0.858, adj=0.348, (0 split)
##       volatile acidity    < 0.4975   to the right, agree=0.830, adj=0.219, (0 split)
##       citric acid         < 0.135    to the left,  agree=0.827, adj=0.205, (0 split)
##       fixed acidity       < 9.05     to the right, agree=0.822, adj=0.181, (0 split)
##
## Node number 2: 1129 observations,    complexity param=0.065625
##   predicted class=red     expected loss=0.1000886  P(node) =0.2171571
##     class counts:  1016   113
##    probabilities: 0.900 0.100
##   left son=4 (1015 obs) right son=5 (114 obs)
##   Primary splits:
##       chlorides        < 0.0465   to the right, improve=149.71360, (0 missing)
##       density          < 0.993295 to the right, improve=111.50150, (0 missing)
##       sulphates        < 0.41     to the right, improve= 84.80043, (0 missing)
##       residual sugar   < 1.15     to the right, improve= 56.77907, (0 missing)
##       volatile acidity < 0.275    to the right, improve= 47.69885, (0 missing)
##   Surrogate splits:
##       density          < 0.99265  to the right, agree=0.948, adj=0.482, (0 split)
##       sulphates        < 0.41     to the right, agree=0.931, adj=0.316, (0 split)
##       residual sugar   < 1.35     to the right, agree=0.928, adj=0.289, (0 split)
##       volatile acidity < 0.205    to the right, agree=0.913, adj=0.140, (0 split)
##       pH               < 2.915    to the right, agree=0.903, adj=0.044, (0 split)
##
## Node number 3: 4070 observations,    complexity param=0.06171875
##   predicted class=whites  expected loss=0.06486486  P(node) =0.7828429
##     class counts:   264  3806
##    probabilities: 0.065 0.935
##   left son=6 (430 obs) right son=7 (3640 obs)
##   Primary splits:
##       chlorides            < 0.0675   to the right, improve=223.07390, (0 missing)
##       volatile acidity     < 0.4875   to the right, improve=168.58930, (0 missing)
##       fixed acidity        < 8.55     to the right, improve= 49.20029, (0 missing)
##       total sulfur dioxide < 92.5     to the left,  improve= 45.40790, (0 missing)
##       density              < 0.995835 to the right, improve= 39.36579, (0 missing)
##   Surrogate splits:
##       volatile acidity < 0.5625   to the right, agree=0.914, adj=0.191, (0 split)
##       fixed acidity    < 9.85     to the right, agree=0.899, adj=0.044, (0 split)
##       sulphates        < 0.985    to the right, agree=0.897, adj=0.028, (0 split)
##       density          < 1.002415 to the right, agree=0.895, adj=0.005, (0 split)
##       pH               < 2.78     to the left,  agree=0.895, adj=0.005, (0 split)
##
## Node number 4: 1015 observations
##   predicted class=red     expected loss=0.0137931  P(node) =0.1952299
##     class counts:  1001    14
##    probabilities: 0.986 0.014
##
## Node number 5: 114 observations
##   predicted class=whites  expected loss=0.1315789  P(node) =0.02192729
##     class counts:    15    99
##    probabilities: 0.132 0.868
##
## Node number 6: 430 observations,    complexity param=0.06171875
##   predicted class=red     expected loss=0.4534884  P(node) =0.08270821
##     class counts:   235   195
##    probabilities: 0.547 0.453
##   left son=12 (234 obs) right son=13 (196 obs)
##   Primary splits:
##       volatile acidity     < 0.415    to the right, improve=87.00182, (0 missing)
##       density              < 0.995215 to the right, improve=82.59426, (0 missing)
##       fixed acidity        < 7.05     to the right, improve=81.00140, (0 missing)
##       total sulfur dioxide < 153.5    to the left,  improve=75.01408, (0 missing)
##       sulphates            < 0.495    to the right, improve=73.61071, (0 missing)
##   Surrogate splits:
##       density             < 0.99545  to the right, agree=0.756, adj=0.464, (0 split)
##       fixed acidity       < 7.175    to the right, agree=0.747, adj=0.444, (0 split)
##       sulphates           < 0.505    to the right, agree=0.737, adj=0.423, (0 split)
##       free sulfur dioxide < 34.5     to the left,  agree=0.693, adj=0.327, (0 split)
##       residual sugar      < 1.85     to the right, agree=0.663, adj=0.260, (0 split)
##
## Node number 7: 3640 observations
##   predicted class=whites  expected loss=0.007967033  P(node) =0.7001346
##     class counts:    29  3611
##    probabilities: 0.008 0.992
##
## Node number 12: 234 observations,    complexity param=0.0203125
##   predicted class=red     expected loss=0.1623932  P(node) =0.04500866
##     class counts:   196    38
##    probabilities: 0.838 0.162
##   left son=24 (208 obs) right son=25 (26 obs)
##   Primary splits:
##       total sulfur dioxide < 157      to the left,  improve=41.04274, (0 missing)
##       residual sugar       < 8.2      to the left,  improve=25.98340, (0 missing)
##       pH                   < 3.115    to the right, improve=20.78310, (0 missing)
##       density              < 0.99498  to the right, improve=13.30084, (0 missing)
##       fixed acidity        < 6.85     to the right, improve=10.64811, (0 missing)
##   Surrogate splits:
##       residual sugar < 8.2      to the left,  agree=0.936, adj=0.423, (0 split)
##       pH             < 3.065    to the right, agree=0.906, adj=0.154, (0 split)
##       fixed acidity  < 6.3      to the right, agree=0.893, adj=0.038, (0 split)
##       citric acid    < 0.62     to the left,  agree=0.893, adj=0.038, (0 split)
##
## Node number 13: 196 observations,    complexity param=0.015625
##   predicted class=whites  expected loss=0.1989796  P(node) =0.03769956
##     class counts:    39   157
##    probabilities: 0.199 0.801
##   left son=26 (32 obs) right son=27 (164 obs)
##   Primary splits:
##       total sulfur dioxide < 89       to the left,  improve=28.79057, (0 missing)
##       fixed acidity        < 7.175    to the right, improve=21.92039, (0 missing)
##       pH                   < 3.255    to the right, improve=18.21463, (0 missing)
##       sulphates            < 0.575    to the right, improve=16.76245, (0 missing)
##       density              < 0.99583  to the right, improve=15.31731, (0 missing)
##   Surrogate splits:
##       fixed acidity < 7.95     to the right, agree=0.862, adj=0.156, (0 split)
##       density       < 0.99033  to the left,  agree=0.852, adj=0.094, (0 split)
##       alcohol       < 13.1     to the right, agree=0.847, adj=0.063, (0 split)
##
## Node number 24: 208 observations
##   predicted class=red     expected loss=0.05769231  P(node) =0.04000769
##     class counts:   196    12
##    probabilities: 0.942 0.058
##
## Node number 25: 26 observations
##   predicted class=whites  expected loss=0  P(node) =0.005000962
##     class counts:     0    26
##    probabilities: 0.000 1.000
##
## Node number 26: 32 observations
##   predicted class=red     expected loss=0.1875  P(node) =0.00615503
##     class counts:    26     6
##    probabilities: 0.812 0.188
##
## Node number 27: 164 observations
##   predicted class=whites  expected loss=0.07926829  P(node) =0.03154453
##     class counts:    13   151
##    probabilities: 0.079 0.921
```

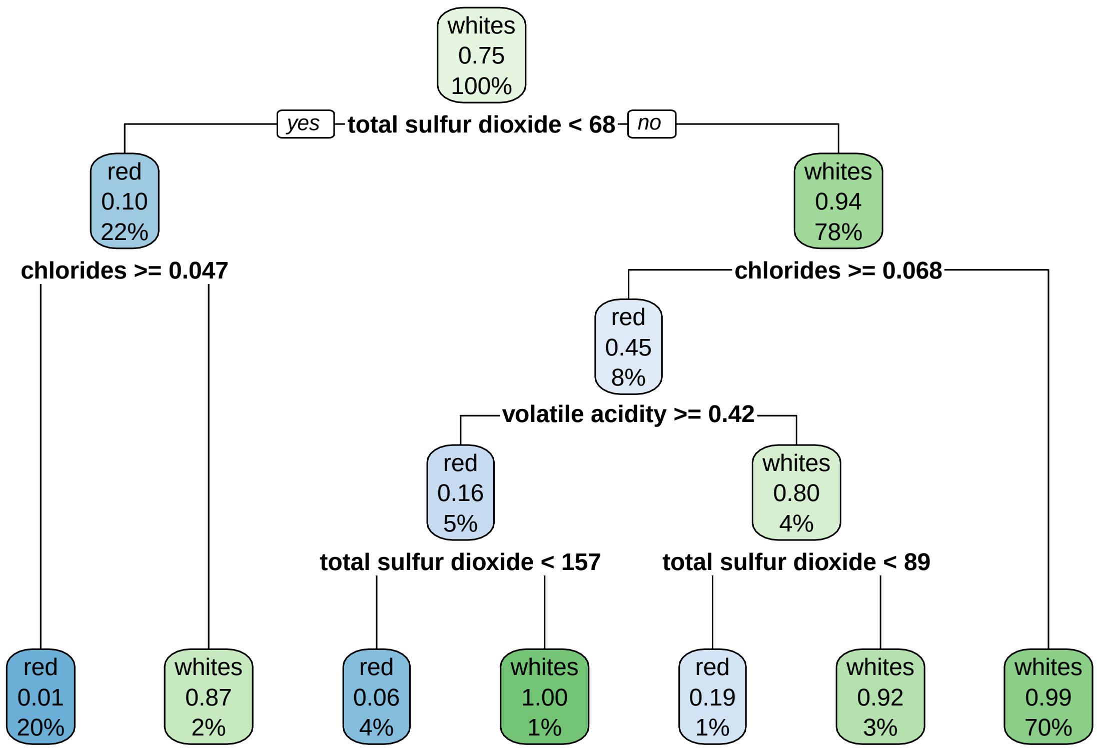
```
## [1] 0.01694915
```

This error rate is also pretty good, but not as good as LDA.

It does however come with this nice visualization and it also reports variable importance metrics. If I was purely going by

```
## Warning: Bad 'data' field in model 'call' (expected a data.frame or a matrix).
## To silence this warning:
```

```
##     Call rpart.plot with roundint=FALSE,
##     or rebuild the rpart model with model=TRUE.
```



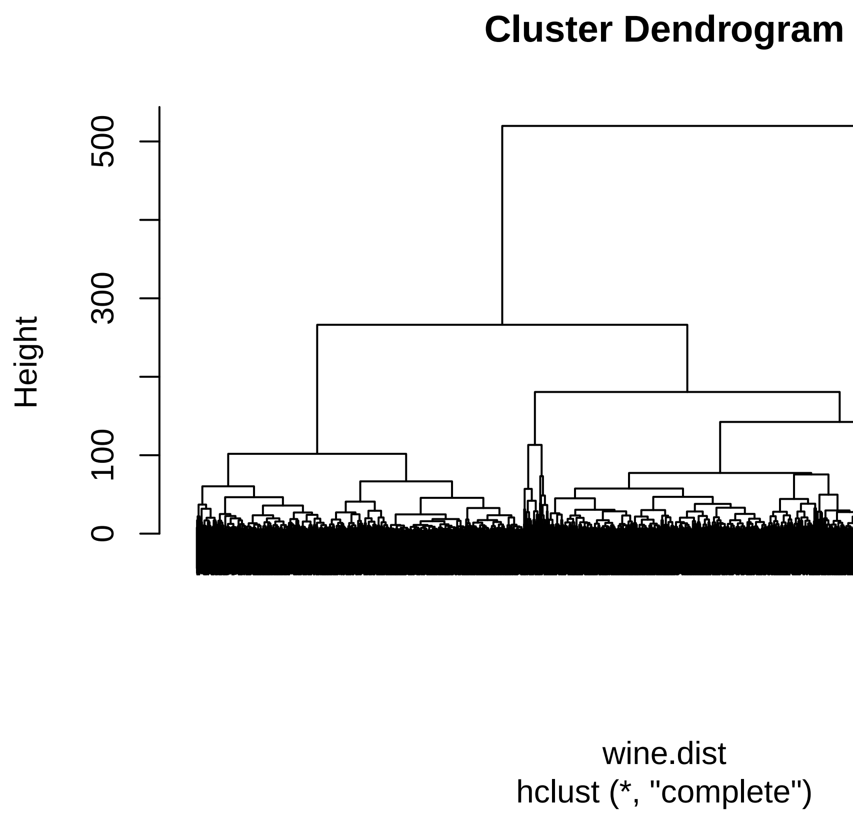
Part c.

Now let's do some classification with k-means!

Trying with both scaling and not scaling the data first.

```
##
##       red whites
##    1 1581   2716
##    2   18   2182
```

Now with hierarchical clustering, again both scaling and not scaling the data beforehand.



```
##
## wine.hier.labels  red whites
##                1 1465    808
##                2  132   3985
##                3    2    104
##                4    0      1
```

Both approaches have a decent amount of error, but I think since we know there are two types of wine, we should be able to use k-means clustering. Doing k-mean clustering with k=2 and scaling the data beforehand seems to produce the best grouping.

I think k-means on the scaled data definitely performs the best.

Part 2

Part a.

For this part, we want to see if there is a difference in the mean vectors for red wines of different qualities. We can do this using MANOVA using the `manova()` function in R!

	Df	Pillai	approx F	num Df	den Df	Pr(>F)
quality	1	0.3606	81.35	11	1587	1.791e-145
Residuals	1597	NA	NA	NA	NA	NA

The `manova()` function conducts a Pillai test which we only briefly talked about in class. This miniscule p-value tells us that there is strong statistical evidence that the mean vectors are different between wines of different qualities!

	Df	Pillai	approx F	num Df	den Df	Pr(>F)
quality	1	0.2287	42.78	11	1587	9.529e-82
Residuals	1597	NA	NA	NA	NA	NA

Collapsing the groups in Low, Medium, and High Quality wines only shrinks the p-value.

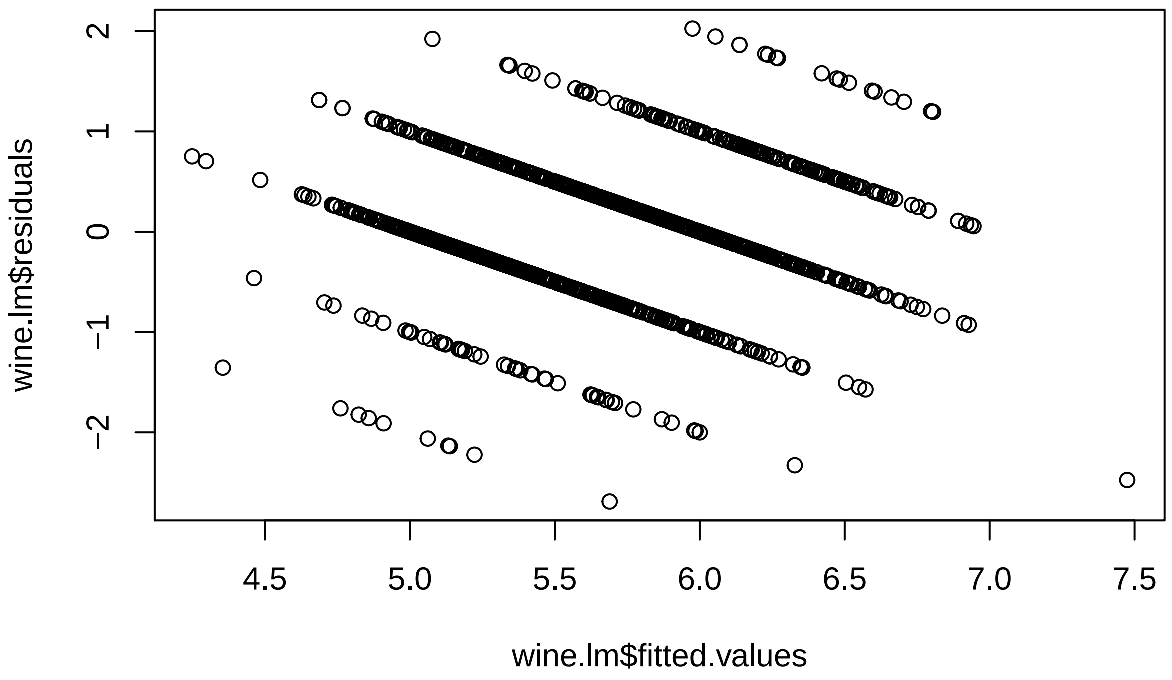
Part b.

```
##   [1] 1322  679 1010  290 1518  243  744 1080 1405  107  281 1203 1364
##  [14] 1437  397  926 1580  965 1260 1133   74  618  892  616  253 1227
##  [27]  726  363 1265  626  248 1349 1258  272  167  664  280  848 1424
##  [40]  562  194  905  301  125 1241 1114 1515 1047  742 1594   71  484
##  [53]  529   27  870  557  773  528 1548  393  209   20 1240  256  282
##  [66]  302   85 1543 1039   79  340 1090 1524  432 1214 1097  994    5
##  [79]  318  825  373  188 1166  781  445 1069  297 1429 1000   87  481
##  [92]   23  127  155  638  414 1313   89  386 1147  842  662  647 1378
## [105]  467   61 1576  704  336   72  331  388  536  688 1151  899  308
## [118]  351  108  794 1501 1324    3  140 1043 1263  369  223  426 1091
## [131]  897  713  937  893  327  306  429   60 1363 1473  596  962  158
## [144]  554  851  611  980 1037 1300  299 1280  109 1176  895  957  354
## [157]  430 1208 1074  831  450   75  169  670  294 1306  174 1019   36
## [170]   26  900  771  465  201  285  833  782 1210 1525  118 1521 1183
## [183]  930 1032 1443  722  629 1335 1274 1377 1442 1399  478   44  566
## [196]  494  472  598  204 1511  995 1124  594 1014  804   41  705  245
## [209]  739  260  410  649  221 1582 1446  815 1407 1245  716  917  317
## [222]  966  752 1188 1068 1426 1588 1257 1264 1509  257 1438 1225  997
## [235]    6  697 1307  690  803  834 1308 1026  493  951 1155 1049  344
## [248]  104 1108  883  974  517  932  922  153 1283  710  278 1595  614
## [261] 1089 1118  765 1520  955  916  333  199  760 1238 1420  552  212
## [274]   19 1262  845 1353  838  181  783  954  753  329  805  229  446
## [287] 1057  200 1181 1115  623  334 1243  124  929  144  574  711  692
## [300]  374  866 1139  648  526  293  241  279  330  471 1136  506 1036
## [313] 1338  389  119 1475 1578   94  233  691  945  615 1419 1441  569
## [326] 1561 1247 1510  258  673 1051 1545  839  878  689 1586 1099  356
## [339] 1401  487  477  137  504  607  621   32  406   46 1484 1052  894
## [352] 1497 1266 1309  593  259  413  986  869 1384 1557  568 1371  888
## [365] 1088 1544  100 1482  835 1357 1017 1174  390  291 1296  323 1317
## [378]  239  316 1187  868  660  485  985  913  289  667 1590  908 1585
## [391] 1472  268  864  314   33  665  763 1385 1132  384   78 1112 1459
## [404]   37 1083  798  793  885  934 1207 1408  178 1593 1232 1452  582
## [417]  745  991  226 1423 1556 1348  619  219 1169 1579  606 1394  168
## [430] 1558  944 1152  123 1167   50  463  364  998  620  578 1293 1116
## [443]   96 1555  592 1321 1469  735 1577 1224  298  717  570  196  527
## [456]  387 1462 1079   92  215  628  800  307 1316 1098  421   25  473
## [469] 1004  539  442  171  542  156  338 1058 1343 1372 1128 1414  353
## [482]  652  740 1498 1284  939  772  887  404   84  668 1333   55   97
## [495] 1569  843 1236 1526  270  177  802  269  567   39  398 1477 1046
## [508]  111 1007 1073  515  583 1490 1461  982  440  589  759 1205  113
##  [521]  524  809 1299  491   51  448 1150  914  721 1396  856 1159 1368
##  [534] 1455 1125 1560 1351 1289 1361  479  106  767 1381  211  377 1229
##  [547]  222  218 1141 1374  490  114  875 1237   81 1425  499  677  630
##  [560] 1082  151   48 1403  811 1581  224 1084  126  964  640 1542  925
##  [573]  579 1538 1040 1100 1382 1467  822  816  956  261 1171  457 1428
##  [586]  208  227  984  958 1231 1070  339   90 1175 1027 1042 1218  921
##  [599]  541  370  850 1254  796 1246  644 1278 1359 1559 1350  288 1514
##  [612]  263  556 1234  266   10 1451  513 1373  346 1440 1033 1433  941
##  [625] 1532 1223  906 1314 1547  840   91  508  509  683 1217 1272  468
##  [638]  786 1519  576 1267  447  255 1512  186   54  641  584  419 1493
##  [651]  172  881   76 1550  368  175  687 1339  852 1273  189  283   18
##  [664] 1130  973 1221 1529  706 1506 1589  220 1270  372 1516 1285 1565
##  [677]  464  535  439  743  931 1304 1172   30  543  720  836 1113 1178
##  [690] 1478  633  861 1400  183  967  813 1418 1571 1328  378   86 1126
##  [703]  275 1342 1465 1291  538 1402 1369 1315 1566  600 1253  936 1230
##  [716]  309 1294  163  814  650 1485  747  599 1093  305 1259  886  360
##  [729] 1456  938  162  488 1063 1102 1481  769 1092  646  832  657    4
##  [742] 1427 1470  551 1352  121 1087 1360 1138  173  284 1044  979  666
##  [755] 1164  685  943   88  555  210 1095  170  325  693  774  829  335
##  [768]  890  139 1598  768  152  399  422  146  217  996 1504 1468  150
##  [781] 1386 1536  441 1034 1282  999 1332  459  497  787  264 1397  601
##  [794]  559  761 1487  128 1107 1365   16  775  639  703 1142 1281 1474
##  [807]   64 1417 1045  924 1239 1355 1413  785 1415  948  409 1277  198
##  [820] 1154 1486 1103 1406 1395  247  970  807  659  658  812  157  867
##  [833] 1292  750  731  312  734  341 1388 1564  989 1404  276  733  415
##  [846] 1430 1105 1161  993 1018 1009 1570  371  661  748   42  411   77
##  [859] 1035  857 1563  553  723 1499  116 1211  514 1075 1275  655  191
##  [872]  362  758  462 1160  350  533 1012  322  902 1310  549 1085  806
##  [885] 1072 1592 1503 1170  133  501 1393 1345 1145  757 1541 1269  489
##  [898] 1367 1466  580  605  919  149   58  195 1182   38  624  424 1295
##  [911] 1567 1121  971 1496   59  483 1534 1453  425 1549  983  741 1436
##  [924]   62 1111  792  819 1193  452 1591 1375  193  799    8 1048 1460
##  [937]  609  808 1180  161  469  265  516  142   93  863  190 1222  503
##  [950] 1573 1513  714  737  751 1219  563 1001  707  627  428 1356  502
##  [963] 1059  612 1101  507  132  534 1480  952  700 1053 1495  942 1202
##  [976]  790   98   82 1110   65 1599  560  466  112  810   17  912 1297
##  [989]  988  518  694  238  602 1123 1062 1268  273  898 1134  784 1204
## [1002] 1086  846  240  228  166  510 1387 1366  367  903 1148  332  321
## [1015]  873   29  235  828 1362  214 1383  236 1416  830  820 1464  101
## [1028] 1024  587   31  572   12 1109  102  634  461 1186  577  992 1435
## [1041]  456  531  320   95  632 1206  548  328 1030  635 1271   67 1162
## [1054]  724 1235  453  591 1507  244  375  408 1117  451  498    1  928
## [1067]  818 1165  500  202 1527  110 1168  147 1197  680  631 1290  313
## [1080]  935  862  182  337  326  841  558 1077 1106    2  987 1319  821
## [1093]  249  653  412 1013  495  643   21  981 1431 1517   66 1347  837
## [1106]  874  795  232 1409  540  197 1137  405 1354  216  595  564 1358
## [1119]  940 1163  311  105 1479  901  696  482 1064  345  449  315 1311
## [1132]  672 1531 1008  120  365  904  923  701  254  187  590  709  823
## [1145]  978  896  251  129 1003 1318  402 1015 1131  359  277  230  789
## [1158] 1213  349  849   73  523  854  876 1376 1071  674  234  343 1200
## [1171]   45 1330 1552  381  271  732 1233 1454 1248  972   70 1528  920
## [1184] 1568  486  654 1421 1173 1199  676  610 1392 1508 1158  438 1305
## [1197]  909   35  135  877 1562 1191  250  403 1135  738 1094 1198  395
## [1210] 1252  546 1522  755  391   63  645  779  537 1242   57  138  959
## [1223]  475  292  669  727 1439  492 1054  431  237  407 1398   15  625
## [1236]  379  300  730 1192  185 1286 1201  865 1329  969  736  520 1505
## [1249]  675  358    7 1539 1016  961 1448 1005 1179 1320 1029  746 1331
## [1262]  444  933  608 1325   47  949  565  148  817   34  134  879  225
## [1275] 1597  454 1530  797  418
```

Come up with a rule to predict wine quality based on the other 11 variables.

For this problem, I would first consider linear regression.

```
##
## Call:
## lm(formula = quality ~ ., data = reds[, -13])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.68911 -0.36652 -0.04699  0.45202  2.02498
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             2.197e+01  2.119e+01   1.036   0.3002
## `fixed acidity`         2.499e-02  2.595e-02   0.963   0.3357
## `volatile acidity`     -1.084e+00  1.211e-01  -8.948  < 2e-16 ***
## `citric acid`          -1.826e-01  1.472e-01  -1.240   0.2150
## `residual sugar`        1.633e-02  1.500e-02   1.089   0.2765
## chlorides              -1.874e+00  4.193e-01  -4.470 8.37e-06 ***
## `free sulfur dioxide`   4.361e-03  2.171e-03   2.009   0.0447 *
## `total sulfur dioxide` -3.265e-03  7.287e-04  -4.480 8.00e-06 ***
## density                -1.788e+01  2.163e+01  -0.827   0.4086
## pH                     -4.137e-01  1.916e-01  -2.159   0.0310 *
## sulphates               9.163e-01  1.143e-01   8.014 2.13e-15 ***
## alcohol                 2.762e-01  2.648e-02  10.429  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.648 on 1587 degrees of freedom
## Multiple R-squared:  0.3606, Adjusted R-squared:  0.3561
## F-statistic: 81.35 on 11 and 1587 DF,  p-value: < 2.2e-16
```



```
## [1] 3164.277
```

The adjusted R^2 isn't very good here. Also, not all of the variables seem to be important.

We can try ordinal regression as well.

```
##
## Re-fitting to get Hessian

## Call:
## polr(formula = quality ~ ., data = reds[, -13])
##
## Coefficients:
##                             Value Std. Error t value
## `fixed acidity`           0.10240   0.051209   2.000
## `volatile acidity`       -3.41794   0.400103  -8.543
## `citric acid`            -0.80494   0.462371  -1.741
## `residual sugar`          0.07617   0.038210   1.993
## chlorides                -5.17121   1.354371  -3.818
## `free sulfur dioxide`     0.01392   0.006767   2.057
## `total sulfur dioxide`   -0.01119   0.002360  -4.744
## density                 -48.92546   0.974499 -50.206
## pH                       -0.98472   0.496900  -1.982
## sulphates                 2.86724   0.358017   8.009
## alcohol                   0.85611   0.059355  14.424
```

```
##
## Intercepts:
##     Value    Std. Error t value
## 3|4 -48.8787   0.9979   -48.9791
## 4|5 -46.9597   0.9959   -47.1537
## 5|6 -43.2452   0.9988   -43.2964
## 6|7 -40.3898   1.0111   -39.9450
## 7|8 -37.3837   1.0409   -35.9135
##
## Residual Deviance: 3074.928
## AIC: 3106.928
```

Fits a little better (smaller AIC).

```
##
## Re-fitting to get Hessian
```

```
## Call:
## polr(formula = quality ~ ., data = reds.train[, -13])
##
## Coefficients:
##                             Value Std. Error t value
## `fixed acidity`           0.10764   0.057129   1.884
## `volatile acidity`       -3.22872   0.435025  -7.422
## `citric acid`            -0.78728   0.511999  -1.538
## `residual sugar`          0.09583   0.041388   2.315
## chlorides                -5.67351   1.449540  -3.914
## `free sulfur dioxide`     0.01369   0.007530   1.818
## `total sulfur dioxide`   -0.01130   0.002583  -4.376
## density                 -82.07375   1.069986 -76.705
## pH                       -0.99725   0.548702  -1.817
## sulphates                 2.89684   0.390994   7.409
## alcohol                   0.84134   0.066848  12.586
##
## Intercepts:
##     Value    Std. Error t value
## 3|4 -81.8164   1.0956   -74.6763
## 4|5 -79.9473   1.0936   -73.1072
## 5|6 -76.3393   1.0970   -69.5860
## 6|7 -73.5245   1.1093   -66.2803
## 7|8 -70.5223   1.1403   -61.8432
##
## Residual Deviance: 2502.672
## AIC: 2534.672
```

```
## [1] 0.609375
```

Our ordinal regression model got 59.38% of the observations correct.

Let's try kNN…

```
##
## knn.reds  3   4   5   6   7   8
##          3  0  0  0  0  0  0
##          4  0  0  2  0  0  0
##          5  1  3 88 39  8  0
##          6  0  5 51 73  9  1
##          7  0  0  5 19 15  1
##          8  0  0  0  0  0  0
```

```
## [1] 0.55
```

For our test set, the kNN classifier only got 46.25% of the observations correct.

Finally, let's try a CART model!

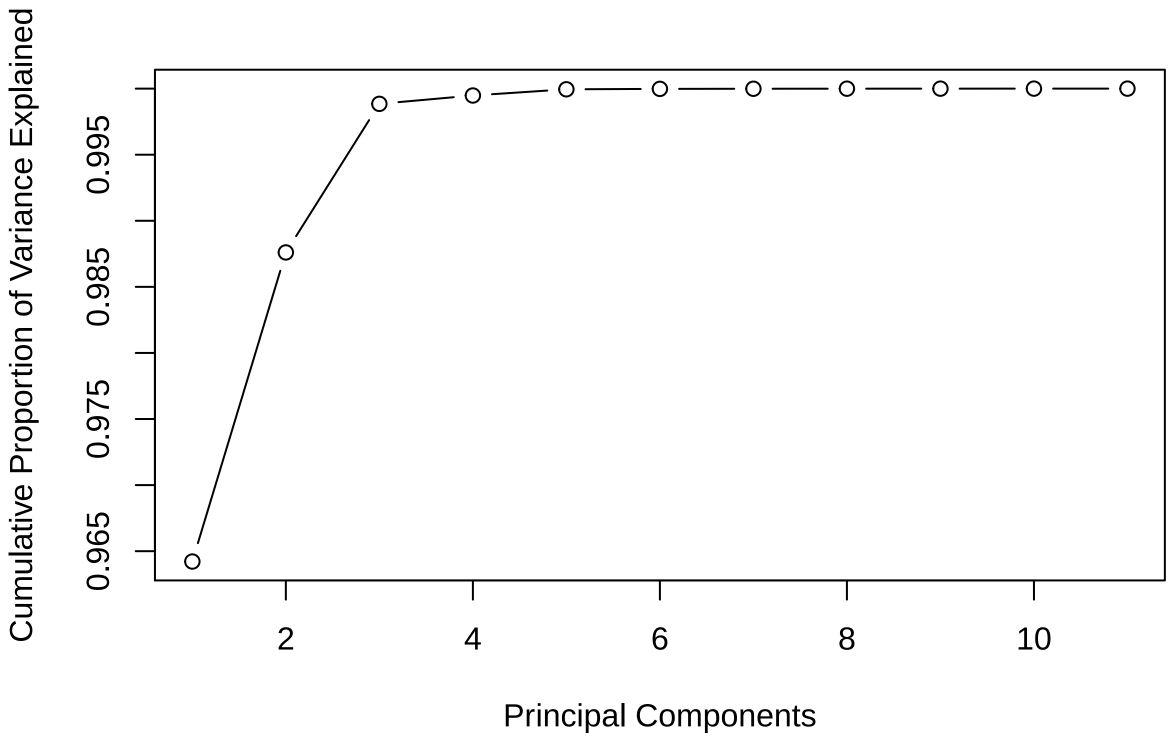
```
## [1] 0.553125
```

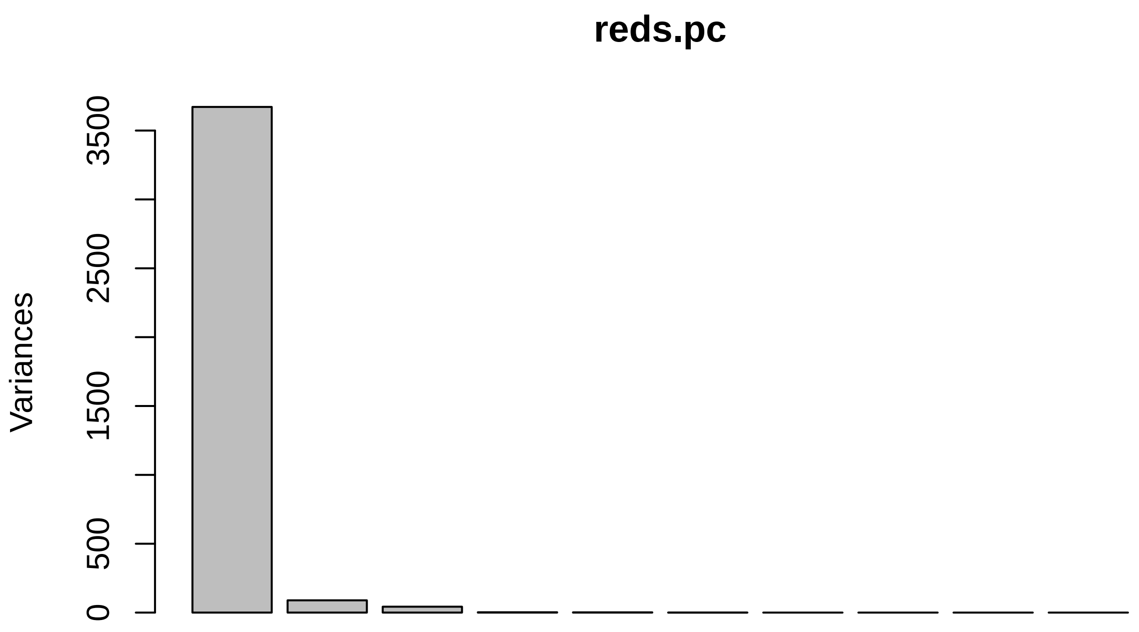
The CART model classified around 56% of the wines correctly.

I would use the ordinal regression model then since it had the highest prediction model of the three models we tested.

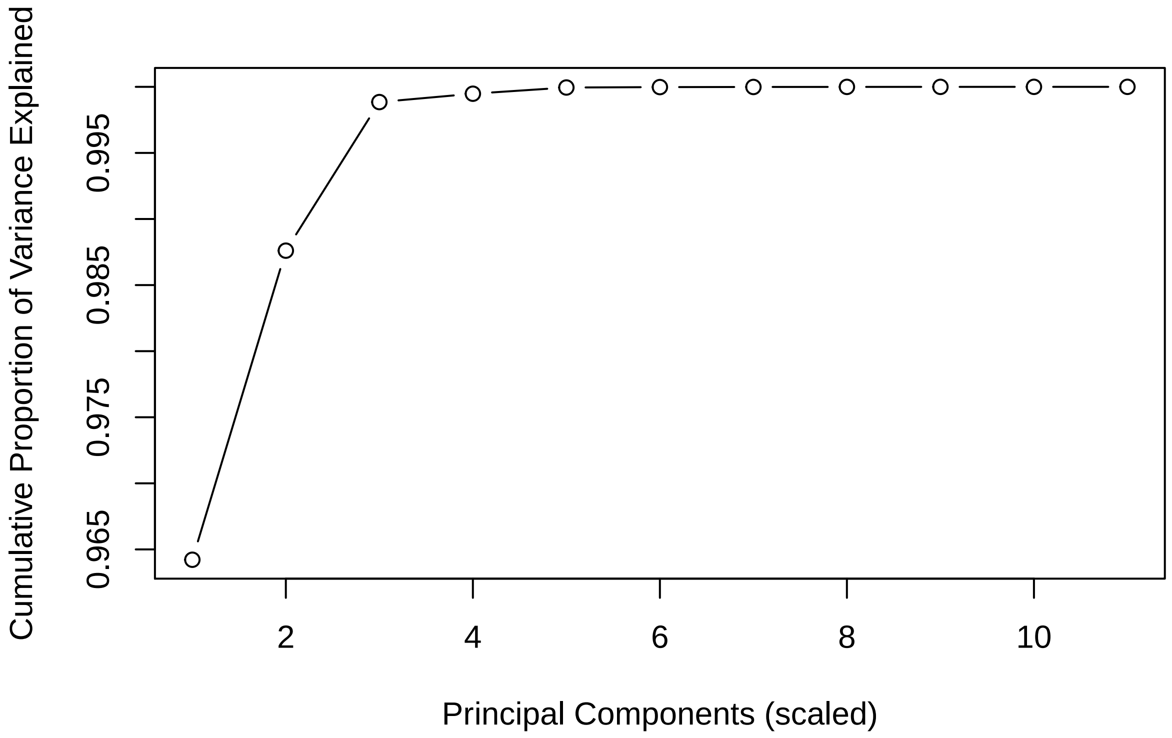
Part c.

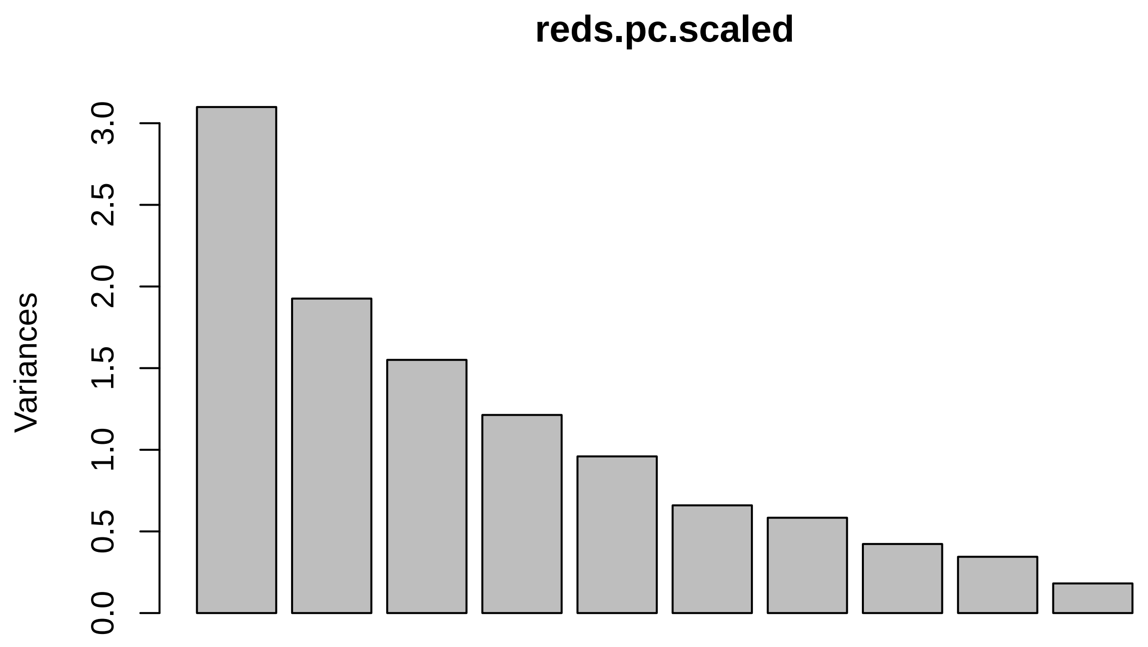
Let's do some PCA on these red wines and see what we get!





This plot shows that the first two principal components explain almost all of the variation in the data with the first PC explaining over 95% of the variance alone.


Cumulative Proportion of Variance Explained
0.965
0.975
0.985
0.995
2
4
6
8
10
Principal Components (scaled)



Centering and scaling makes it so the principal conponents explain more equal shares of total variance. This is probably not what we want however – we want most of the variance to be explained by just a few PCs.

Let's try a PC regression and see if it better predicts the quality of the wines vs. the other models we had.

The `pls` library has a function to handle PC regression.

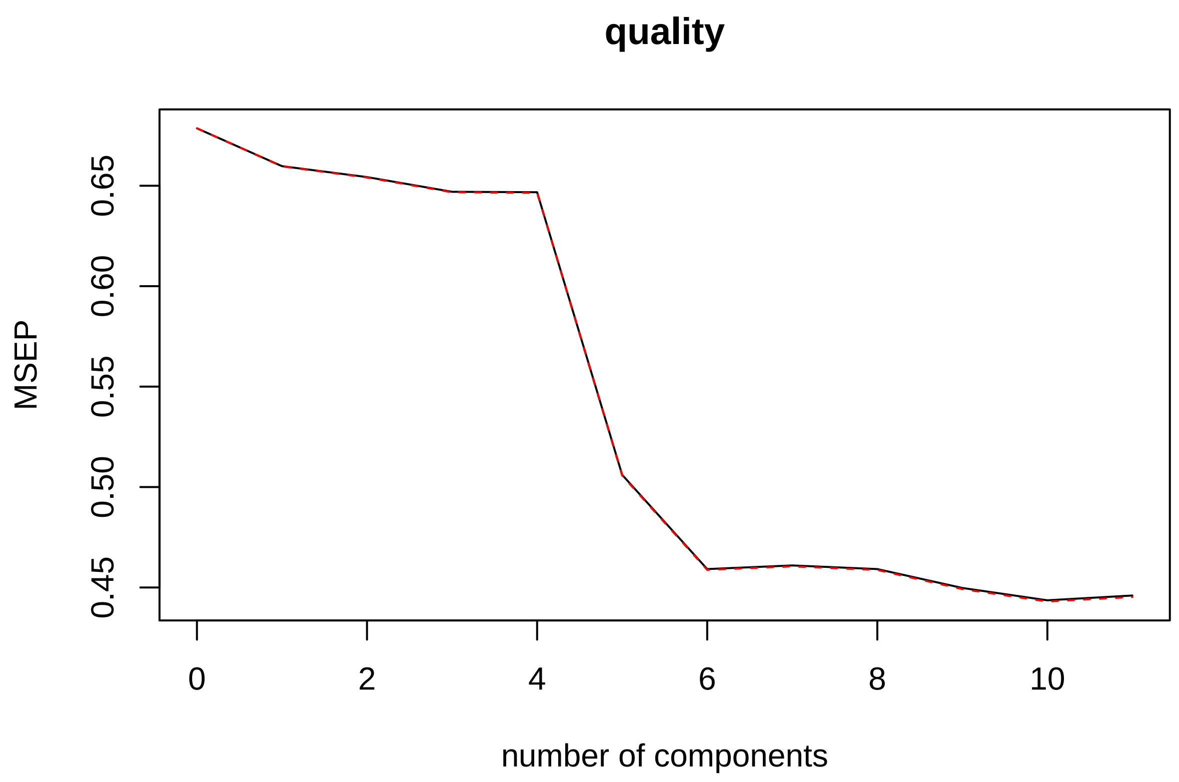
```
## Warning: package 'pls' was built under R version 3.6.1
```

```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    1279 obs. of  12 variables:
##  $ fixed acidity       : num  5 8.3 9.6 11.6 6.5 7.7 11.6 7.9 8 7.8 ...
##  $ volatile acidity    : num  0.74 0.78 0.5 0.42 0.53 0.58 0.41 0.3 0.5 0.41 ...
##  $ citric acid         : num  0 0.1 0.36 0.53 0.06 0.1 0.58 0.68 0.39 0.68 ...
##  $ residual sugar      : num  1.2 2.6 2.8 3.3 2 1.8 2.8 8.3 2.6 1.7 ...
##  $ chlorides           : num  0.041 0.081 0.116 0.105 0.063 0.102 0.096 0.05 0.082 0.467 ...
##  $ free sulfur dioxide : num  16 45 26 33 29 28 25 37.5 12 18 ...
##  $ total sulfur dioxide: num  46 87 55 98 44 109 101 278 46 69 ...
##  $ density             : num  0.993 0.998 0.997 1.001 0.995 ...
##  $ pH                  : num  4.01 3.48 3.18 3.2 3.38 3.08 3.13 3.01 3.43 3.08 ...
##  $ sulphates           : num  0.59 0.53 0.68 0.95 0.83 0.49 0.53 0.51 0.62 1.31 ...
##  $ alcohol             : num  12.5 10 10.9 9.2 10.3 9.8 10 12.3 10.7 9.3 ...
##  $ quality             : Ord.factor w/ 6 levels "3"<"4"<"5"<"6"<..: 4 3 3 3 4 4 3 5 4 3 ...

## Data:    X dimension: 1279 11
##  Y dimension: 1279 1
## Fit method: svdpc
## Number of components considered: 11
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV          0.8238   0.8123   0.8089   0.8044   0.8042   0.7114   0.6776
## adjCV       0.8238   0.8122   0.8088   0.8042   0.8040   0.7111   0.6773
##        7 comps  8 comps  9 comps  10 comps  11 comps
## CV      0.6790   0.6776   0.6706    0.6661    0.6678
## adjCV   0.6786   0.6773   0.6702    0.6656    0.6673
##
## TRAINING: % variance explained
##          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps
## X         94.884   99.508   99.752   99.907    99.99   100.00   100.00
## quality    3.027    4.031    5.265    5.561    26.36    33.25    33.26
##          8 comps  9 comps  10 comps  11 comps
## X         100.00   100.00    100.00    100.00
## quality    33.68    35.18     36.21     36.24
```



```
##
##        3   4   5   6   7   8
##    5   0   2  32  11   2   0
##    6   1   6 114 120  30   2
```

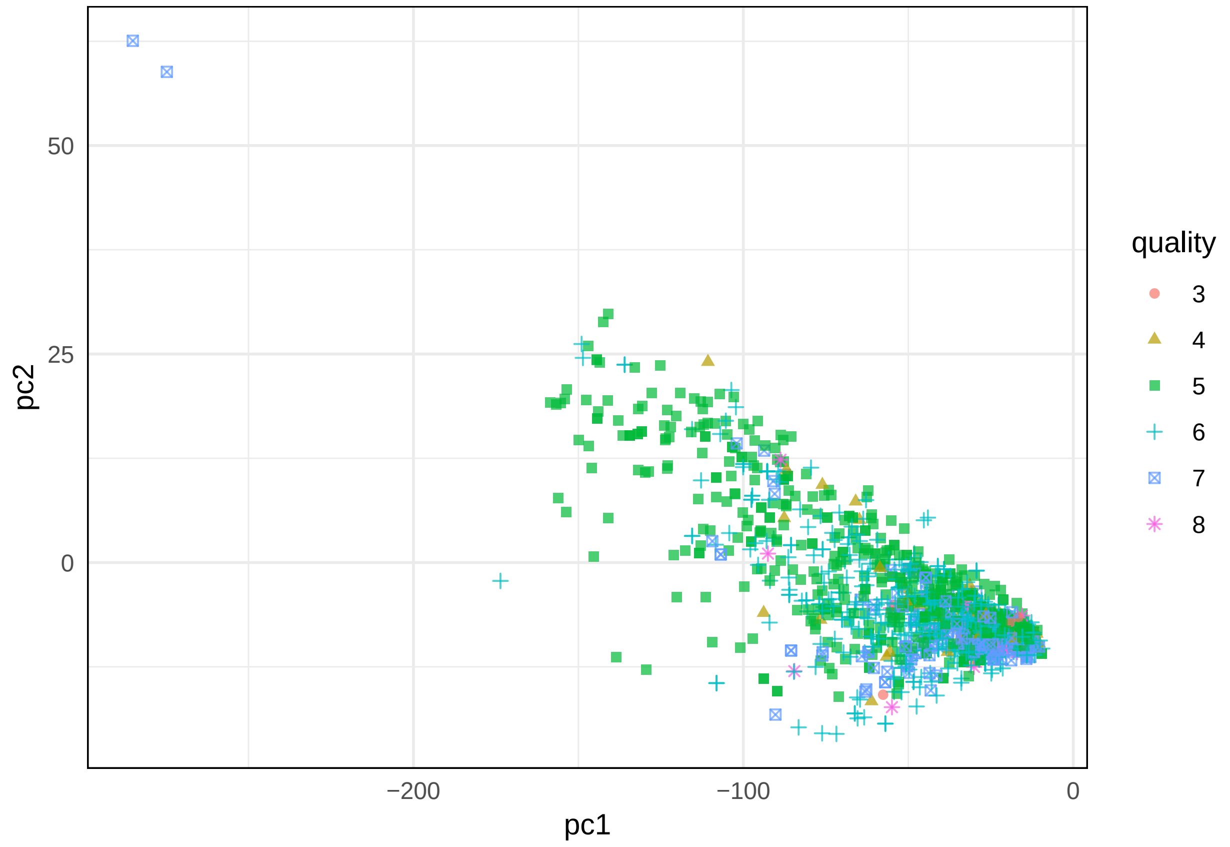
Hm. Doesn't seem to be working and the `pcr()` function can't handle an ordered factor response. Also, the number of principle components that are being selected seems to be around 6. This is more PCs than is being asked for this question.

Let's try a different approach.

Let's instead use the loadings to create a plot of the first 2 PC variables and see what we can do with those.

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

Let's plot these new PCs!



Hmm… interesting? Don't see any distinct clustering within the PCs for quality… Is this better than before? Let's find out. Let's revisit the models we considered before, but now with this reduced dimensionality PC dataset.

```
##
## Re-fitting to get Hessian
```

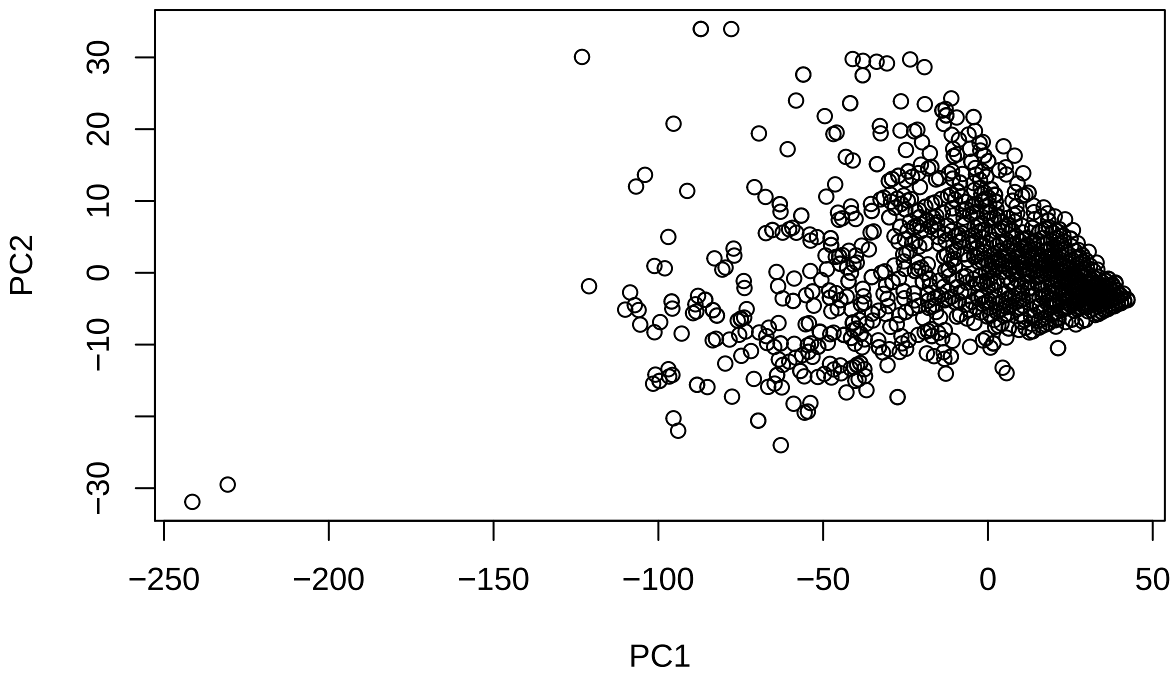
```
## Call:
## polr(formula = quality ~ ., data = reds.pc.train)
```

```
##
## Coefficients:
##          Value Std. Error  t value
## pc1  4.054e-05   0.002435  0.01665
## pc2 -6.592e-02   0.009820 -6.71258
##
## Intercepts:
##     Value    Std. Error t value
## 3|4  -4.8657   0.3696   -13.1650
## 4|5  -3.0434   0.2105   -14.4556
## 5|6   0.0887   0.1668     0.5313
## 6|7   2.1493   0.1790    12.0084
## 7|8   4.7590   0.2985    15.9410
##
## Residual Deviance: 2980.738
## AIC: 2994.738
```

```
## [1] 0.178125
```

Whoa, the predictive power dropped hard using just the first two PC variables!

Note: Easier way to get the scores of the PCs is just to use this:


PC2
PC1