Red Wines - upper

Katie, Rita, and Chang

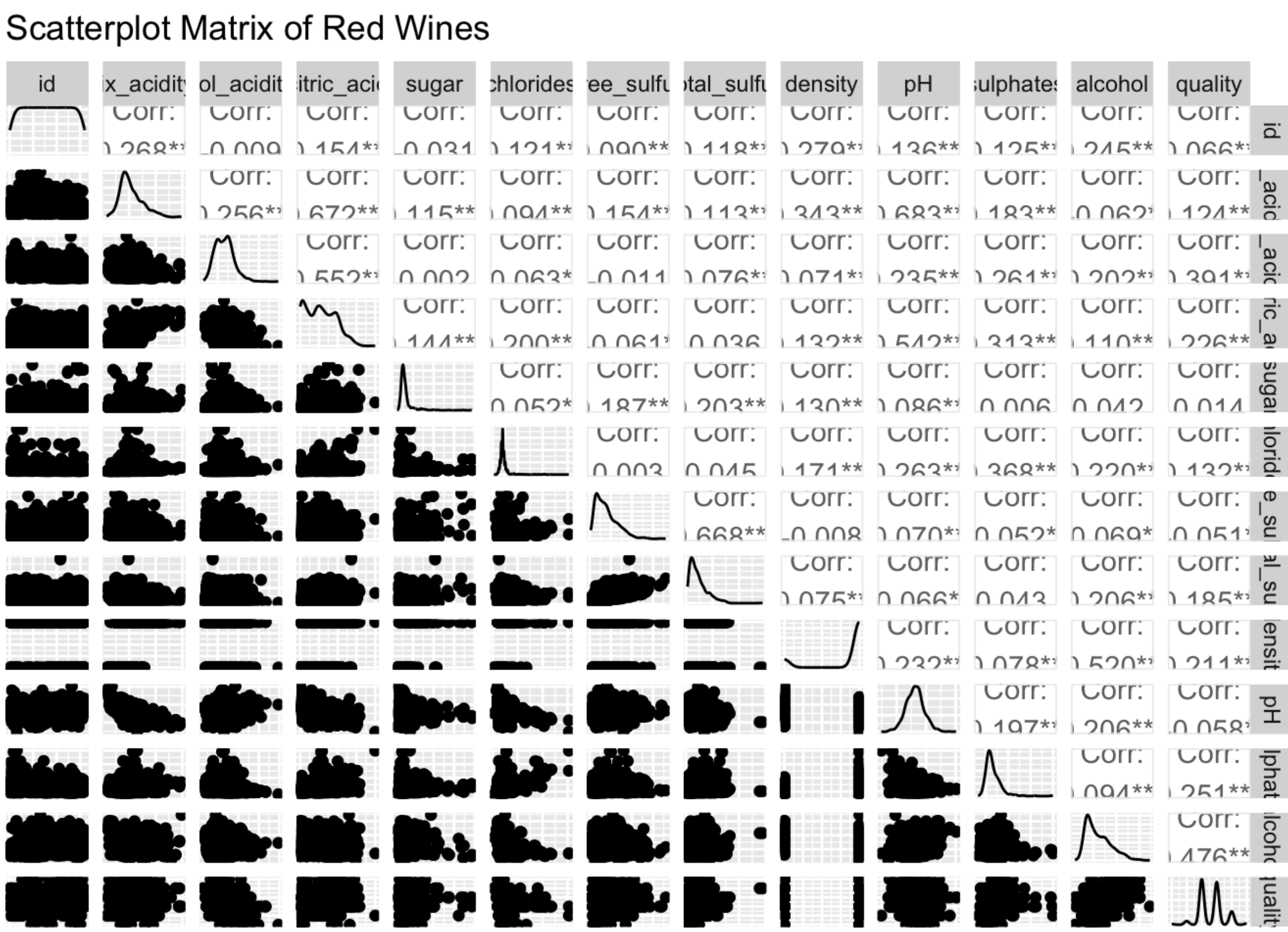
2023-11-01

Scatterplot Matrix

```r
library("GGally")
ggpairs(red, axisLabels = "none",
        title = "Scatterplot Matrix of Red Wines")
```

```r
# corr codes
```

Scatterplot Matrix



Create Binary Dependent Variable

```
red$highquality = factor((red$quality >= 6))
red$highquality <- as.integer(as.logical(red$highquality))
```

Create Test and Training Data

```
library("caTools")
set.seed = 100
split = sample.split(red$highquality, SplitRatio = 0.6)
train = subset(red, split == TRUE)
test = subset(red, split == FALSE)
print(dim(train)); print(dim(test))
```

```
## [1] 959  14
```

```
## [1] 640  14
```

Descriptive Statistics

```
library("Rmisc")
```

```
## Loading required package: lattice
```

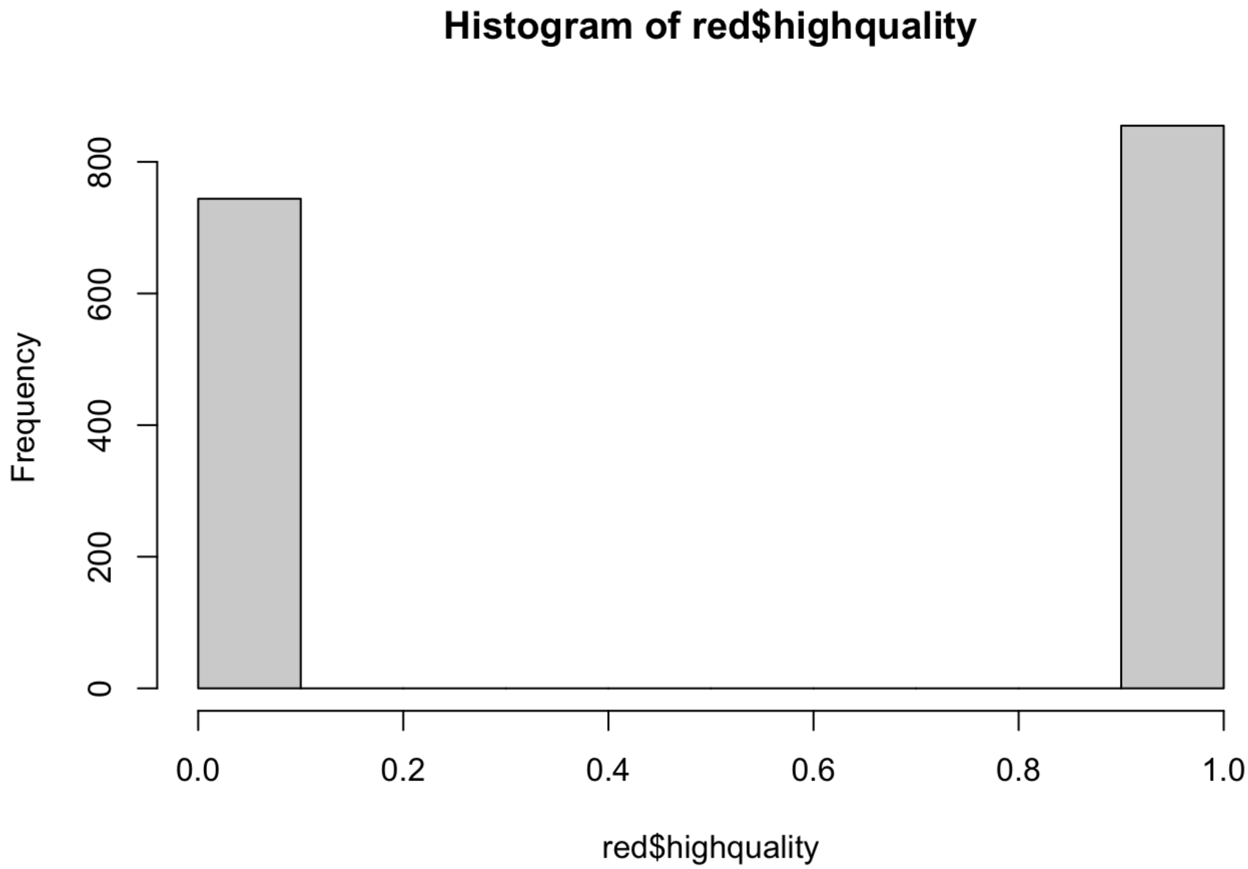
```
## Loading required package: plyr
```

```
sum = summary(red)
sum
```

```
##        id          fix_acidity     vol_acidity      citric_acid
##  Min.   :   1.0   Min.   : 4.60   Min.   :0.1200   Min.   :0.000
##  1st Qu.: 400.5   1st Qu.: 7.10   1st Qu.:0.3900   1st Qu.:0.090
##  Median : 800.0   Median : 7.90   Median :0.5200   Median :0.260
##  Mean   : 800.0   Mean   : 8.32   Mean   :0.5284   Mean   :0.271
##  3rd Qu.:1199.5   3rd Qu.: 9.20   3rd Qu.:0.6400   3rd Qu.:0.420
##  Max.   :1599.0   Max.   :15.90   Max.   :1.5800   Max.   :1.000
##      sugar          chlorides         free_sulfur     total_sulfur
##  Min.   : 0.900   Min.   :0.01000   Min.   : 1.00   Min.   :  6.00
##  1st Qu.: 1.900   1st Qu.:0.07000   1st Qu.: 7.00   1st Qu.: 22.00
##  Median : 2.200   Median :0.08000   Median :14.00   Median : 38.00
##  Mean   : 2.539   Mean   :0.08787   Mean   :15.87   Mean   : 46.47
##  3rd Qu.: 2.600   3rd Qu.:0.09000   3rd Qu.:21.00   3rd Qu.: 62.00
##  Max.   :15.500   Max.   :0.61000   Max.   :72.00   Max.   :289.00
##     density             pH          sulphates         alcohol
##  Min.   :0.9900   Min.   :2.740   Min.   :0.3300   Min.   : 8.40
##  1st Qu.:1.0000   1st Qu.:3.210   1st Qu.:0.5500   1st Qu.: 9.50
##  Median :1.0000   Median :3.310   Median :0.6200   Median :10.20
##  Mean   :0.9985   Mean   :3.311   Mean   :0.6581   Mean   :10.42
##  3rd Qu.:1.0000   3rd Qu.:3.400   3rd Qu.:0.7300   3rd Qu.:11.10
##  Max.   :1.0000   Max.   :4.010   Max.   :2.0000   Max.   :14.90
##     quality       highquality
##  Min.   :3.000   Min.   :0.0000
##  1st Qu.:5.000   1st Qu.:0.0000
##  Median :6.000   Median :1.0000
##  Mean   :5.636   Mean   :0.5347
##  3rd Qu.:6.000   3rd Qu.:1.0000
##  Max.   :8.000   Max.   :1.0000
```

Plot high quality vs low quality distribution

```
hist (red$highquality)
```



Random Forest

```
library("randomForest")
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

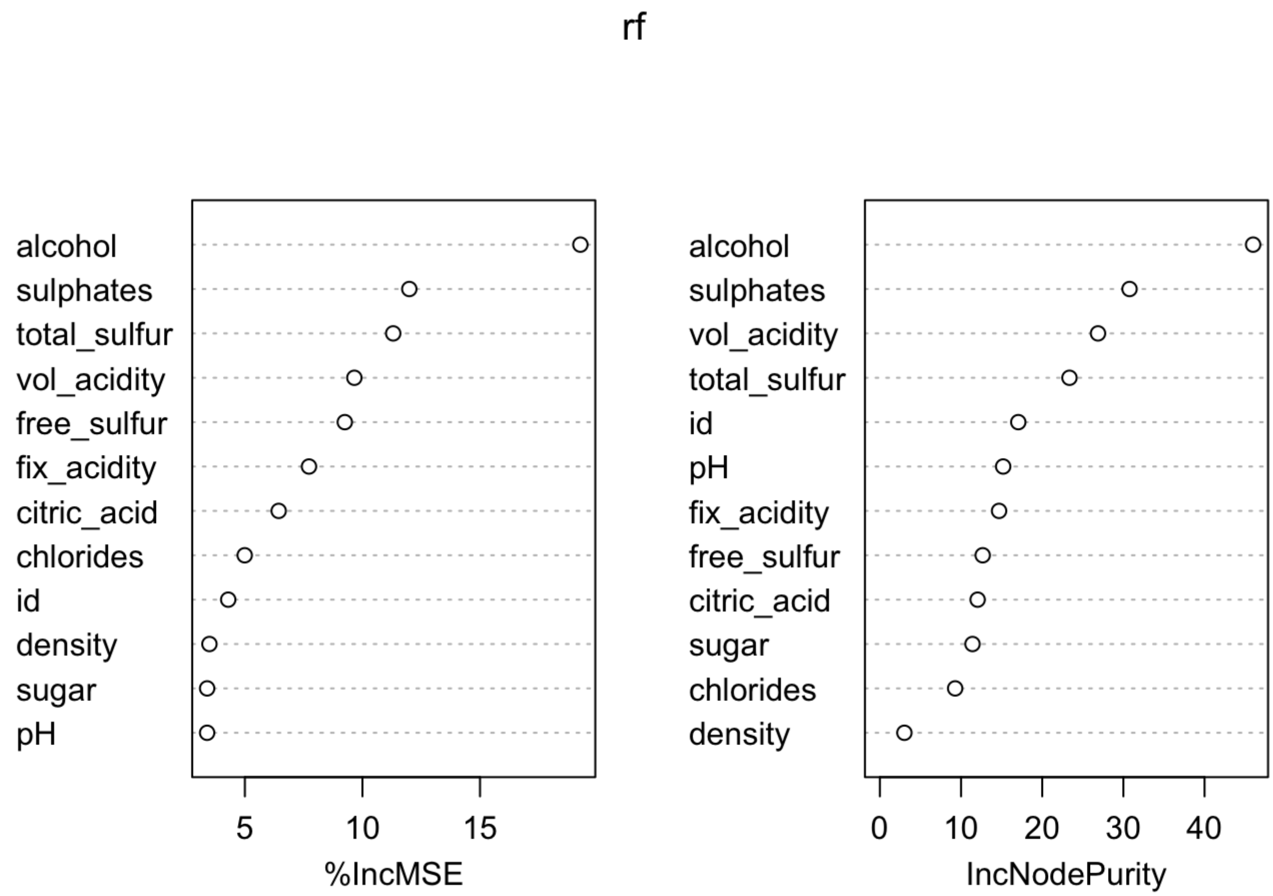
```
library("caret")
library("e1071")
library("rpart")

rf <- randomForest(highquality ~ . - quality, data = train, mtry = 4, importance = TRUE,
ntree = 50, na.action = na.omit)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
print(rf)
```

```
##
## Call:
##  randomForest(formula = highquality ~ . - quality, data = train,      mtry = 4, impor
tance = TRUE, ntree = 50, na.action = na.omit)
##                Type of random forest: regression
##                      Number of trees: 50
## No. of variables tried at each split: 4
##
##           Mean of squared residuals: 0.1482476
##                     % Var explained: 40.41
```

```
varImpPlot(rf)
```




```
# predictions on test set
set.seed(100)
predictTest = predict(rf, newdata = test, type = "response")

# confusion matrix on test set
table(test$highquality, predictTest >= 0.5)
```

```
##
##       FALSE TRUE
##    0    221   77
##    1     66  276
```

```
520/nrow(test)
```

```
## [1] 0.8125
```

```
# the model is accurate 81.3 percent of the time
```

Random Forest Model

```
# Logit
randomforestmodlogit <- glm(highquality ~ alcohol + sulphates + total_sulfur + vol_acidi
ty, data = red, family = "binomial"(link = "logit"))
summary(randomforestmodlogit)
```

```
##
## Call:
## glm(formula = highquality ~ alcohol + sulphates + total_sulfur +
##     vol_acidity, family = binomial(link = "logit"), data = red)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1638  -0.8675   0.3076   0.8629   2.3262
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -8.588813   0.795118 -10.802  < 2e-16 ***
## alcohol       0.927362   0.069268  13.388  < 2e-16 ***
## sulphates     2.059047   0.365976   5.626 1.84e-08 ***
## total_sulfur -0.011976   0.001924  -6.225 4.83e-10 ***
## vol_acidity  -3.083277   0.364832  -8.451  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2209.0  on 1598  degrees of freedom
## Residual deviance: 1684.2  on 1594  degrees of freedom
## AIC: 1694.2
##
## Number of Fisher Scoring iterations: 4
```

```
# Cloglog
randomforestmodcloglog <- glm(highquality ~ alcohol + sulphates + total_sulfur + vol_aci
dity, data = red, family = "binomial"(link = "cloglog"))
summary(randomforestmodcloglog)
```

```
##
## Call:
## glm(formula = highquality ~ alcohol + sulphates + total_sulfur +
##     vol_acidity, family = binomial(link = "cloglog"), data = red)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.5006  -0.9020   0.2185   0.9295   2.0506
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.958517   0.478252 -10.368  < 2e-16 ***
## alcohol       0.505807   0.038543  13.123  < 2e-16 ***
## sulphates     1.324184   0.221318   5.983 2.19e-09 ***
## total_sulfur -0.009109   0.001364  -6.679 2.41e-11 ***
## vol_acidity  -2.022997   0.238813  -8.471  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2209  on 1598  degrees of freedom
## Residual deviance: 1701  on 1594  degrees of freedom
## AIC: 1711
##
## Number of Fisher Scoring iterations: 7
```

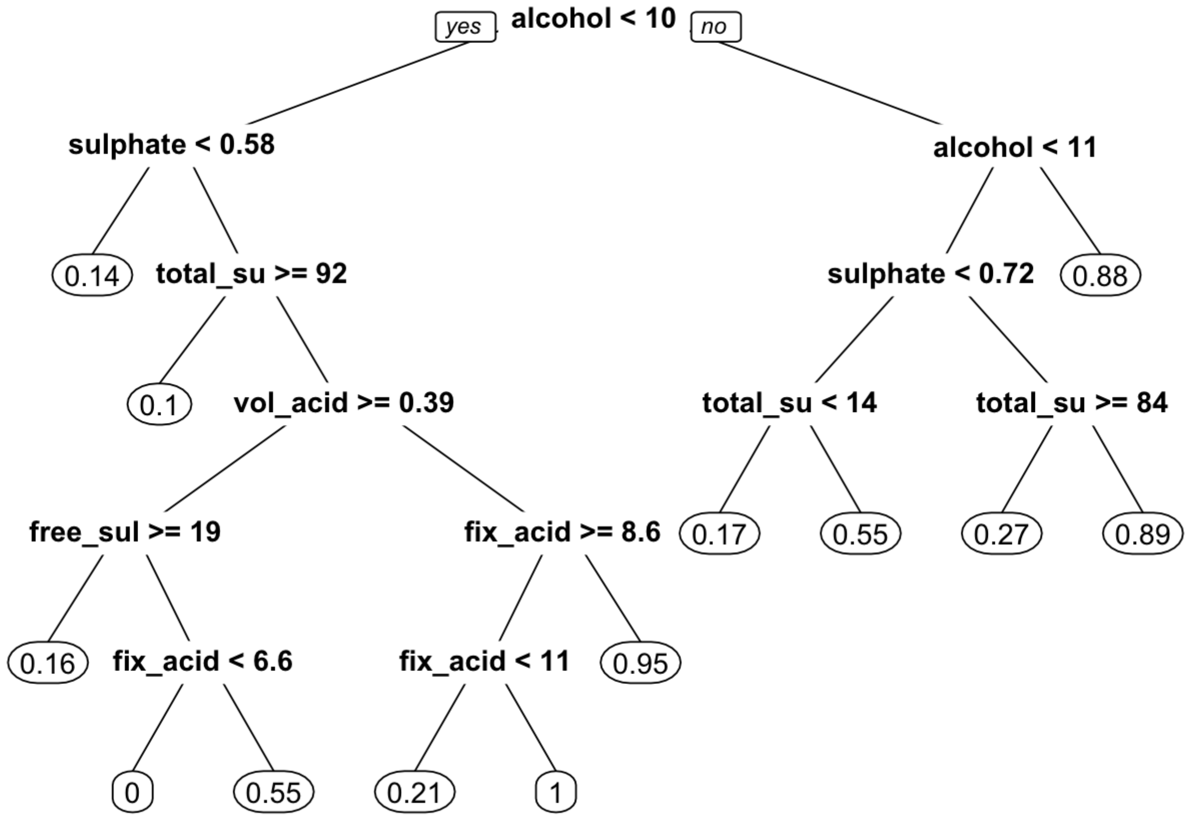
```
# The logit model performed better with a lower AIC value
```

Cart

```
library("caret")
library("e1071")
library("rpart")
library("rpart.plot")

cartmodel = rpart(highquality ~ . - quality, data = train)
print(cartmodel)
```

```
## n= 959
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 959 238.5798000 0.5349322
##    2) alcohol< 9.95 411  84.9635000 0.2919708
##      4) sulphates< 0.575 179  21.5083800 0.1396648 *
##      5) sulphates>=0.575 232  56.0991400 0.4094828
##       10) total_sulfur>=91.5 40   3.6000000 0.1000000 *
##       11) total_sulfur< 91.5 192  47.8697900 0.4739583
##         22) vol_acidity>=0.385 149  35.8389300 0.4026846
##           44) free_sulfur>=18.5 43   5.8604650 0.1627907 *
##           45) free_sulfur< 18.5 106  26.5000000 0.5000000
##             90) fix_acidity< 6.55 9   0.0000000 0.0000000 *
##             91) fix_acidity>=6.55 97  24.0412400 0.5463918 *
##         23) vol_acidity< 0.385 43   8.6511630 0.7209302
##           46) fix_acidity>=8.6 24   5.9583330 0.5416667
##             92) fix_acidity< 11.35 14   2.3571430 0.2142857 *
##             93) fix_acidity>=11.35 10   0.0000000 1.0000000 *
##           47) fix_acidity< 8.6 19   0.9473684 0.9473684 *
##    3) alcohol>=9.95 548 111.1588000 0.7171533
##      6) alcohol< 11.15 340  80.5264700 0.6147059
##       12) sulphates< 0.715 227  56.7400900 0.5066079
##         24) total_sulfur< 13.5 24   3.3333330 0.1666667 *
##         25) total_sulfur>=13.5 203  50.3054200 0.5467980 *
##       13) sulphates>=0.715 113  15.8053100 0.8318584
##         26) total_sulfur>=84 11   2.1818180 0.2727273 *
##         27) total_sulfur< 84 102   9.8137250 0.8921569 *
##      7) alcohol>=11.15 208  21.2307700 0.8846154 *
```

```
prp(cartmodel)
```




```
# predictions on test set
set.seed(100)
predictTest = predict(cartmodel, newdata = test, type = "matrix")

# confusion matrix on test set
table(test$highquality, predictTest >= 0.5)
```

```
##
##     FALSE TRUE
##   0   146  152
##   1    45  297
```

Cart Model

```
# Logit
cartmodlogit <- glm(highquality ~ alcohol + sulphates + total_sulfur + fix_acidity, data
= red, family = "binomial"(link = "logit"))
summary(cartmodlogit)
```

```
##
## Call:
## glm(formula = highquality ~ alcohol + sulphates + total_sulfur +
##     fix_acidity, family = binomial(link = "logit"), data = red)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.3737  -0.9154   0.3562   0.8762   2.0206
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -12.172146   0.828396 -14.694  < 2e-16 ***
## alcohol        0.989178   0.068276  14.488  < 2e-16 ***
## sulphates      2.587844   0.370028   6.994 2.68e-12 ***
## total_sulfur  -0.011171   0.001895  -5.895 3.75e-09 ***
## fix_acidity    0.109461   0.035511   3.082  0.00205 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2209.0  on 1598  degrees of freedom
## Residual deviance: 1752.5  on 1594  degrees of freedom
## AIC: 1762.5
##
## Number of Fisher Scoring iterations: 4
```

```
# Cloglog
cartmodcloglog <- glm(highquality ~ alcohol + sulphates + total_sulfur + fix_acidity, da
ta = red, family = "binomial"(link = "cloglog"))
summary(cartmodcloglog)
```

```
##
## Call:
## glm(formula = highquality ~ alcohol + sulphates + total_sulfur +
##     fix_acidity, family = binomial(link = "cloglog"), data = red)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.7058  -0.9408   0.3075   0.9490   1.9387
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -6.835907   0.481268 -14.204  < 2e-16 ***
## alcohol       0.542953   0.037720  14.394  < 2e-16 ***
## sulphates     1.639060   0.217233   7.545 4.52e-14 ***
## total_sulfur -0.009315   0.001384  -6.732 1.67e-11 ***
## fix_acidity   0.027351   0.021284   1.285    0.199
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2209.0  on 1598  degrees of freedom
## Residual deviance: 1777.5  on 1594  degrees of freedom
## AIC: 1787.5
##
## Number of Fisher Scoring iterations: 18
```

```
# The logit model performed better with the lower AIC value
```

Compare best logit model with AIC

```
library("AICcmodavg")
```

```
##
## Attaching package: 'AICcmodavg'
```

```
## The following object is masked from 'package:randomForest':
##
##     importance
```

```
models <- list(randomforestmodlogit, cartmodlogit)
mod.names <- c('RandomForest', 'Cart')
aictab(cand.set = models, modnames = mod.names)
```

```
##
## Model selection based on AICc:
##
##               K    AICc Delta_AICc AICcWt Cum.Wt      LL
## RandomForest 5 1694.21       0.00      1      1 -842.09
## Cart         5 1762.56      68.35      0      1 -876.26
```

```
# The random forest logit model performed the best
```

Compare best model with BIC

```
library("flexmix")
BIC(randomforestmodlogit)
```

```
## [1] 1721.058
```

```
BIC(randomforestmodcloglog)
```

```
## [1] 1737.845
```

```
BIC(cartmodlogit)
```

```
## [1] 1789.404
```

```
BIC(cartmodcloglog)
```

```
## [1] 1814.418
```

```
# The random forest logit model performed the best
```

Confusion matrix for random forest logit model

```
confusionred = predict(randomforestmodlogit, newdata = red, type = "response")

# confusion matrix on test set
table(red$highquality, confusionred >= 0.5)
```

```
##
##      FALSE TRUE
##    0   548  196
##    1   216  639
```

Predictions for random forest logit model

```
pred_test <- predict(randomforestmodlogit, test, type = "response")

pred_test
```

```
##          6          11          14          16          18          19          25 
## 0.22573317 0.18064756 0.69723806 0.13867752 0.56805457 0.20109254 0.49785778 
##         26          28          30          33          34          35          36 
## 0.42819628 0.58076699 0.38187954 0.17799414 0.15765907 0.37517558 0.31575154 
##         40          41          43          44          46          49          50 
## 0.60541307 0.60541307 0.78970651 0.75462869 0.91140600 0.43019840 0.27520137 
##         51          53          56          60          61          64          67 
## 0.30073364 0.43459537 0.24266029 0.37446288 0.35540196 0.22177359 0.36302140 
##         72          75          77          79          81          82          83 
## 0.11856745 0.50539651 0.67706290 0.19429727 0.38434763 0.65378965 0.19907576 
##         84          86          88          89          94          95         100 
## 0.44994570 0.45450681 0.43771981 0.50840606 0.43771981 0.11648871 0.24182616 
##        101         102         103         106         110         112         113 
## 0.41257714 0.59957642 0.24182616 0.17168420 0.05669027 0.20639675 0.21064410 
##        115         116         119         123         126         127         128 
## 0.46170336 0.61534547 0.59589245 0.21410390 0.19469456 0.15237332 0.14973294 
##        132         133         137         138         139         142         143 
## 0.88460563 0.88460563 0.26189520 0.36049701 0.16095337 0.26189520 0.98548676 
##        146         148         150         152         154         155         158 
## 0.08710726 0.29279616 0.65840863 0.86018228 0.24118269 0.44034290 0.43822117 
##        159         160         163         164         167         168         169 
## 0.21501940 0.15834014 0.45821490 0.10323416 0.22409934 0.18953659 0.43015992 
##        171         176         177         179         184         188         189 
## 0.09027691 0.42523959 0.40084611 0.21924830 0.19319740 0.25256465 0.13003255 
##        197         200         202         204         209         212         213 
## 0.41050290 0.29660130 0.32718865 0.33477584 0.27161312 0.23949363 0.67118625 
##        215         218         219         223         224         232         234 
## 0.30759231 0.14689457 0.37301967 0.36725609 0.39670029 0.47608964 0.64457164 
##        239         240         241         243         244         248         250 
## 0.19063712 0.06682702 0.39273703 0.17002266 0.67646997 0.14653731 0.56422747 
##        251         254         258         259         261         269         271 
## 0.70288800 0.11931143 0.18145879 0.71498229 0.45452948 0.50344768 0.57130009 
##        273         274         276         277         278         290         291 
## 0.61403829 0.22637701 0.57130009 0.50344768 0.87483178 0.36131597 0.69463023 
##        294         297         299         302         303         304         308 
## 0.67070411 0.14998798 0.42029514 0.79587338 0.41165107 0.27204195 0.51647504 
##        309         312         313         318         321         322         326 
## 0.51219806 0.16092124 0.25470497 0.29176280 0.68916140 0.22451563 0.32170068 
##        330         331         334         336         337         339         340 
## 0.47496687 0.94054119 0.60570094 0.88542182 0.94917073 0.80821030 0.88983001 
##        341         342         343         344         345         346         349 
## 0.74512435 0.89763321 0.66228068 0.66228068 0.68982583 0.34930907 0.57482872 
##        353         355         356         363         365         367         370 
## 0.31982246 0.73846936 0.86315416 0.40286206 0.44355920 0.44355920 0.97850481 
##        371         372         374         381         385         387         388 
## 0.28282101 0.55754674 0.21115302 0.65714868 0.32025802 0.31578174 0.24745643 
##        389         390         395         396         402         405         407 
## 0.35745073 0.59089106 0.40524750 0.95715624 0.75725600 0.20675079 0.74518550 
##        408         410         411         413         415         421         422 
## 0.76171904 0.37340589 0.29774721 0.13582974 0.16976587 0.79052736 0.69125350 
##        423         424         430         431         433         435         437 
## 0.22216089 0.91967155 0.32784541 0.91967155 0.96051533 0.62568563 0.31324574
```

```
##        443        444        445        446        447        448        454 
## 0.63537526 0.86779419 0.93413308 0.32689043 0.64968242 0.83321168 0.88497153 
##        456        459        462        465        466        468        470 
## 0.95768591 0.88497153 0.29995555 0.55427468 0.74550780 0.98771509 0.19630137 
##        473        474        475        477        479        480        482 
## 0.68760803 0.83494087 0.69215031 0.69504790 0.41915117 0.43048988 0.95378202 
##        485        486        487        494        495        496        501 
## 0.97215159 0.35050201 0.35050201 0.64444930 0.81342325 0.84287574 0.27636155 
##        503        508        509        513        515        516        520 
## 0.88317746 0.36855117 0.40270830 0.56322747 0.87663195 0.18839969 0.69166380 
##        522        524        527        529        531        534        536 
## 0.35179506 0.15801090 0.69166380 0.28289255 0.85077943 0.96809870 0.85077943 
##        538        540        542        547        550        554        555 
## 0.55097625 0.79067776 0.81658345 0.44918649 0.36606459 0.26817772 0.72091694 
##        556        560        565        567        570        571        572 
## 0.72091694 0.92935383 0.92935383 0.20667837 0.86080487 0.87692510 0.86080487 
##        573        574        578        582        584        585        589 
## 0.77494478 0.28595807 0.22627342 0.38002007 0.69766881 0.71570169 0.98243548 
##        590        592        593        597        606        608        611 
## 0.83511782 0.52956620 0.33060442 0.43790682 0.27296251 0.53420556 0.40906300 
##        612        622        623        624        626        629        633 
## 0.46929105 0.17261769 0.30620045 0.92396940 0.38478177 0.31752796 0.63251372 
##        634        636        638        646        649        650        652 
## 0.32737388 0.19060263 0.03693294 0.52329572 0.79606704 0.49913547 0.19901970 
##        657        659        661        662        663        666        668 
## 0.40958829 0.53465925 0.53465925 0.32678756 0.41930315 0.30272887 0.51847411 
##        670        673        676        679        680        686        687 
## 0.51847411 0.01313453 0.63951248 0.15831687 0.50931169 0.61274001 0.19820088 
##        688        690        695        698        704        706        707 
## 0.30853776 0.71405217 0.17339004 0.33118031 0.39592542 0.09514858 0.27290235 
##        713        714        716        718        720        721        722 
## 0.18728926 0.30467669 0.27155586 0.46339361 0.22428109 0.31642036 0.15248206 
##        723        725        726        729        731        733        736 
## 0.50744957 0.36264216 0.49939352 0.43341582 0.39042992 0.17713379 0.11882456 
##        738        741        745        747        749        752        758 
## 0.24236950 0.71932009 0.24216251 0.26780776 0.47090034 0.24423781 0.17874689 
##        759        761        762        763        764        765        768 
## 0.17874689 0.18944864 0.21832154 0.35211810 0.21832154 0.21925919 0.17859090 
##        769        770        771        773        776        778        779 
## 0.19045961 0.23550990 0.19045961 0.09786165 0.16206695 0.49724573 0.66183909 
##        781        784        786        787        792        795        799 
## 0.28239984 0.49407054 0.36376503 0.36376503 0.10866121 0.95029362 0.66715805 
##        802        803        804        805        809        810        813 
## 0.42345371 0.92132982 0.35326525 0.48671568 0.48396705 0.60727563 0.74001422 
##        814        815        821        822        823        824        825 
## 0.86887538 0.84554487 0.32105800 0.98401461 0.43070016 0.43070016 0.59452832 
##        827        829        831        832        835        837        839 
## 0.88097191 0.91803729 0.60268443 0.75438553 0.21302698 0.73172430 0.90163251 
##        840        841        843        847        852        856        859 
## 0.45005102 0.92522550 0.65608836 0.38077515 0.44134957 0.78141884 0.79806178 
##        860        864        872        873        874        878        881 
## 0.81336010 0.19469921 0.53427892 0.53560900 0.89396556 0.68842602 0.40681831 
```

```
##        883        891        897        898        907        909        912 
## 0.94676764 0.57461423 0.95762419 0.54837049 0.61655717 0.57631602 0.85616038 
##        914        915        918        924        925        926        928 
## 0.93641743 0.88685399 0.57254719 0.57254719 0.84029182 0.85851776 0.14108907 
##        932        933        935        937        943        944        946 
## 0.37796683 0.35714165 0.61571145 0.88587548 0.51350024 0.39208642 0.85009985 
##        948        950        951        953        954        958        963 
## 0.96007673 0.96739607 0.96739607 0.89304382 0.94769354 0.75906185 0.42379344 
##        967        970        971        972        975        976        979 
## 0.91290529 0.53399341 0.91601469 0.91601469 0.93910091 0.28346202 0.78896295 
##        980        982        984        986        988        990        991 
## 0.70963694 0.26394962 0.70374096 0.76216515 0.31207909 0.80108051 0.40407434 
##        992        994        995        998       1000       1001       1003 
## 0.32169795 0.32169795 0.24614218 0.91773179 0.89973442 0.88981382 0.94122194 
##       1005       1007       1009       1010       1011       1012       1013 
## 0.55234587 0.94122194 0.93177532 0.67244034 0.94994037 0.72592191 0.13893015 
##       1014       1016       1017       1018       1019       1020       1022 
## 0.33596228 0.89906458 0.95824366 0.90328264 0.90328264 0.62120744 0.91151720 
##       1024       1026       1028       1031       1034       1035       1040 
## 0.94561683 0.31368976 0.53740392 0.77233191 0.54729817 0.18049828 0.80300719 
##       1041       1046       1049       1050       1051       1052       1054 
## 0.24523640 0.79667103 0.80953886 0.78021636 0.60752858 0.67804325 0.97117164 
##       1063       1065       1067       1068       1078       1082       1084 
## 0.87540468 0.75516090 0.92897417 0.89593617 0.82775448 0.37324283 0.89315798 
##       1090       1092       1093       1102       1103       1105       1107 
## 0.59911292 0.87464611 0.80753908 0.88355318 0.74464695 0.92344246 0.95333070 
##       1108       1112       1113       1118       1123       1125       1131 
## 0.94262703 0.89860782 0.83718598 0.67849027 0.87671921 0.80352421 0.59465043 
##       1133       1136       1145       1146       1149       1153       1154 
## 0.98621359 0.90991385 0.55703889 0.81699614 0.88830835 0.32861855 0.78814787 
##       1162       1163       1164       1168       1169       1173       1177 
## 0.68472828 0.93097631 0.34911691 0.95811695 0.89257541 0.94556825 0.38959620 
##       1180       1184       1186       1188       1191       1193       1195 
## 0.83725702 0.18325571 0.83216767 0.83216767 0.93726418 0.96428659 0.16205917 
##       1196       1197       1200       1201       1203       1207       1208 
## 0.36820273 0.18261742 0.18261742 0.43683111 0.90407751 0.84882533 0.44462253 
##       1210       1212       1216       1219       1220       1223       1225 
## 0.89081590 0.33645556 0.86572123 0.75790425 0.80645578 0.19169762 0.77019805 
##       1228       1229       1233       1236       1237       1240       1241 
## 0.43944184 0.95997919 0.35954537 0.70226268 0.38859556 0.76884549 0.14929864 
##       1244       1251       1253       1256       1259       1261       1262 
## 0.22116768 0.61830476 0.33313384 0.50363201 0.69403671 0.44018286 0.37701038 
##       1263       1264       1268       1269       1273       1277       1278 
## 0.38122840 0.18918454 0.90213067 0.37253912 0.71983985 0.91209807 0.35315581 
##       1279       1281       1282       1285       1286       1287       1288 
## 0.13504841 0.57652097 0.57652097 0.72038518 0.64986391 0.95921380 0.90845738 
##       1293       1295       1298       1301       1304       1305       1307 
## 0.93827703 0.59190236 0.87066317 0.87211203 0.81440601 0.08307448 0.23809254 
##       1308       1314       1317       1321       1322       1324       1330 
## 0.58216493 0.50024518 0.79987457 0.27840724 0.79987457 0.82228483 0.21628945 
##       1333       1336       1338       1339       1344       1345       1349 
## 0.44771727 0.89912713 0.33063230 0.33063230 0.57850387 0.76295728 0.23966391 
```

```
##       1352       1357       1358       1359       1361       1363       1364 
## 0.71942809 0.57441114 0.68815611 0.13052325 0.63469844 0.55865303 0.13872049 
##       1365       1372       1373       1377       1378       1380       1382 
## 0.76732852 0.91160867 0.30081941 0.18491208 0.83265348 0.62870483 0.16546771 
##       1383       1388       1391       1399       1401       1405       1409 
## 0.18145955 0.32286340 0.90986800 0.49039723 0.10211360 0.62668380 0.96956379 
##       1417       1420       1421       1425       1428       1441       1442 
## 0.67996830 0.11430274 0.39766821 0.66331164 0.81035058 0.87184058 0.09511377 
##       1443       1444       1446       1450       1451       1457       1459 
## 0.43582198 0.77179526 0.09753535 0.87394441 0.87184058 0.37601586 0.77734136 
##       1461       1462       1463       1466       1468       1469       1472 
## 0.59735007 0.34284216 0.47637617 0.33900665 0.30916495 0.47414501 0.86017441 
##       1473       1474       1476       1479       1482       1489       1492 
## 0.84882869 0.58301194 0.96416743 0.28090696 0.81973890 0.79522301 0.79522301 
##       1493       1495       1496       1497       1503       1505       1506 
## 0.79984815 0.69740812 0.77693701 0.14135590 0.28363547 0.85817699 0.36554744 
##       1508       1513       1515       1517       1522       1526       1530 
## 0.85817699 0.43899218 0.12840124 0.82595385 0.25612152 0.45939419 0.34422384 
##       1531       1532       1536       1537       1542       1543       1545 
## 0.85054796 0.42672777 0.37906001 0.54738050 0.84898279 0.36046967 0.89062172 
##       1546       1548       1552       1553       1555       1556       1558 
## 0.45450879 0.82069018 0.24069329 0.62386686 0.48014133 0.64128386 0.48014133 
##       1559       1560       1563       1568       1570       1571       1573 
## 0.12503776 0.14727854 0.37557065 0.37557065 0.78071517 0.96429949 0.10892466 
##       1574       1575       1576       1581       1586       1589       1591 
## 0.88033989 0.51978506 0.86164481 0.90818659 0.88371638 0.83629523 0.85094463 
##       1596       1597       1598 
## 0.74190964 0.75118218 0.45044094
```

Model Diagnostics

```
accuracy = (548+639)/(548+196+216+639)
accuracy
```

```
## [1] 0.742339
```

```
sensitivity = 639/(639+196)
sensitivity
```

```
## [1] 0.7652695
```

```
specificity = 548/(548+216)
specificity
```

```
## [1] 0.7172775
```

AUC and ROC

```
library("pROC")
```

```
## Type 'citation("pROC")' for a citation.
```

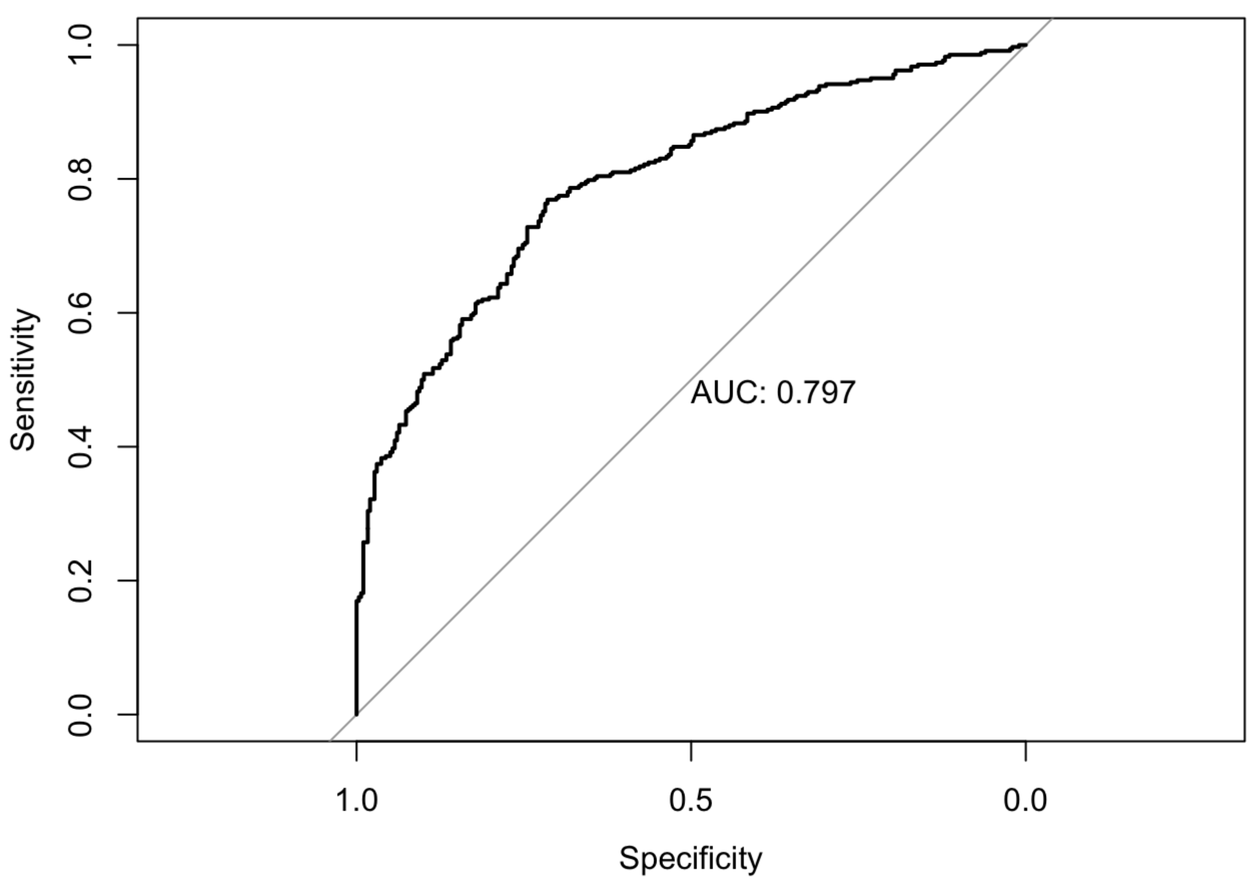
```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
test_prob = predict(randomforestmodlogit, test, type = "response")

test_roc = roc(test$highquality ~ test_prob, plot = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```



```
as.numeric(test_roc$auc)
```

```
## [1] 0.7968425
```

AUC and ROC with just one variable

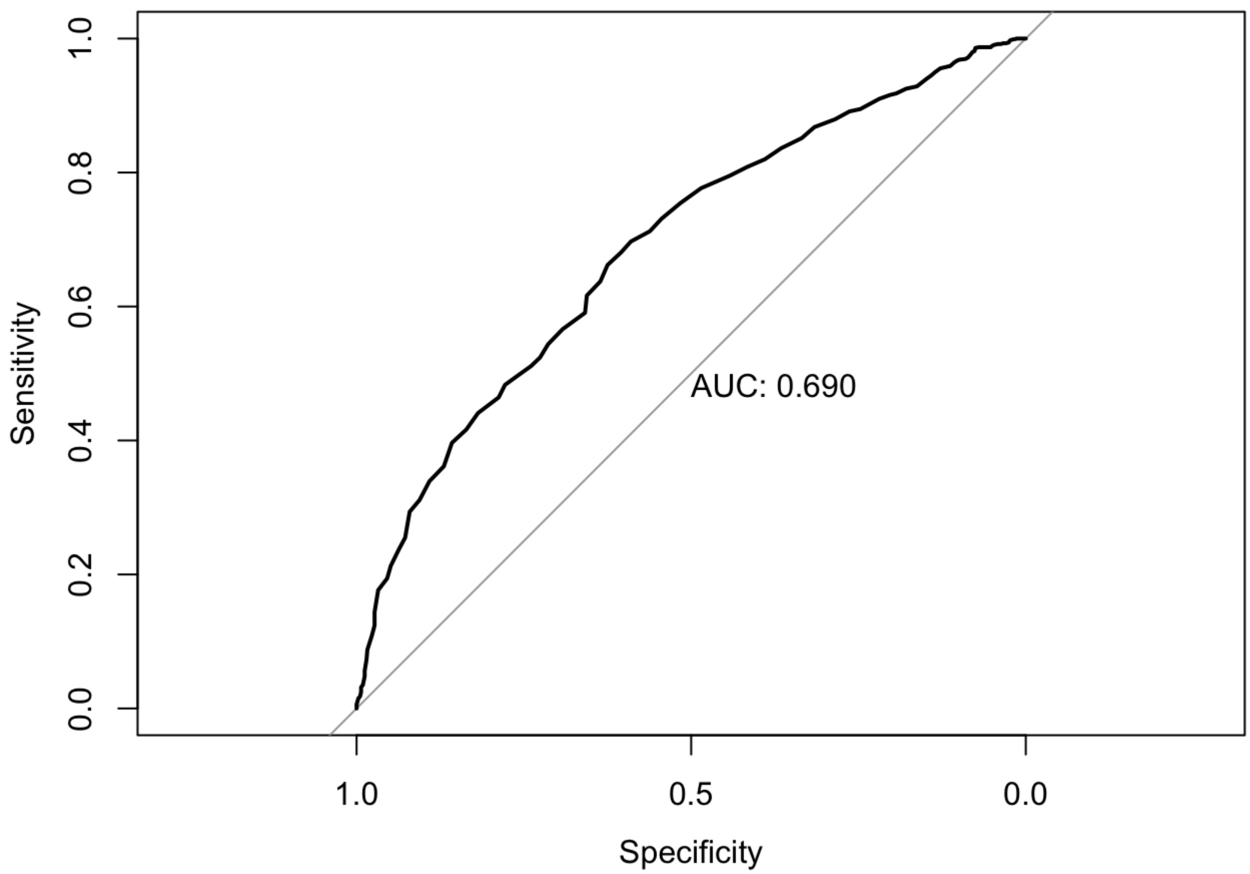
```
simple <- glm(highquality ~ vol_acidity, data = red, family = "binomial"(link = "logi
t"))
summary(simple)
```

```
##
## Call:
## glm(formula = highquality ~ vol_acidity, family = binomial(link = "logit"),
##     data = red)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8697  -1.1148   0.7156   1.0375   2.0349
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.2874     0.1838   12.45   <2e-16 ***
## vol_acidity  -4.0607     0.3334  -12.18   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2209.0  on 1598  degrees of freedom
## Residual deviance: 2033.4  on 1597  degrees of freedom
## AIC: 2037.4
##
## Number of Fisher Scoring iterations: 4
```

```
test_prop1 = predict(simple, red, type = "response")

test_roc1 = roc(red$highquality ~ test_prop1, plot = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
as.numeric(test_roc1$auc)
```

```
## [1] 0.6900011
```