Predicting blood donations

Maximilian Press + Morgan Lawless

Various looks at what you can do with models, hopefully with an emphasis on parametric (GLM) models and the R model object.

Takes a statistical learning point of view on the problem.

Dolph Schluter's R modeling pages are a good resource for general-purpose model fitting. https://www.zoology.ubc.ca/~schluter/R/fit-model/

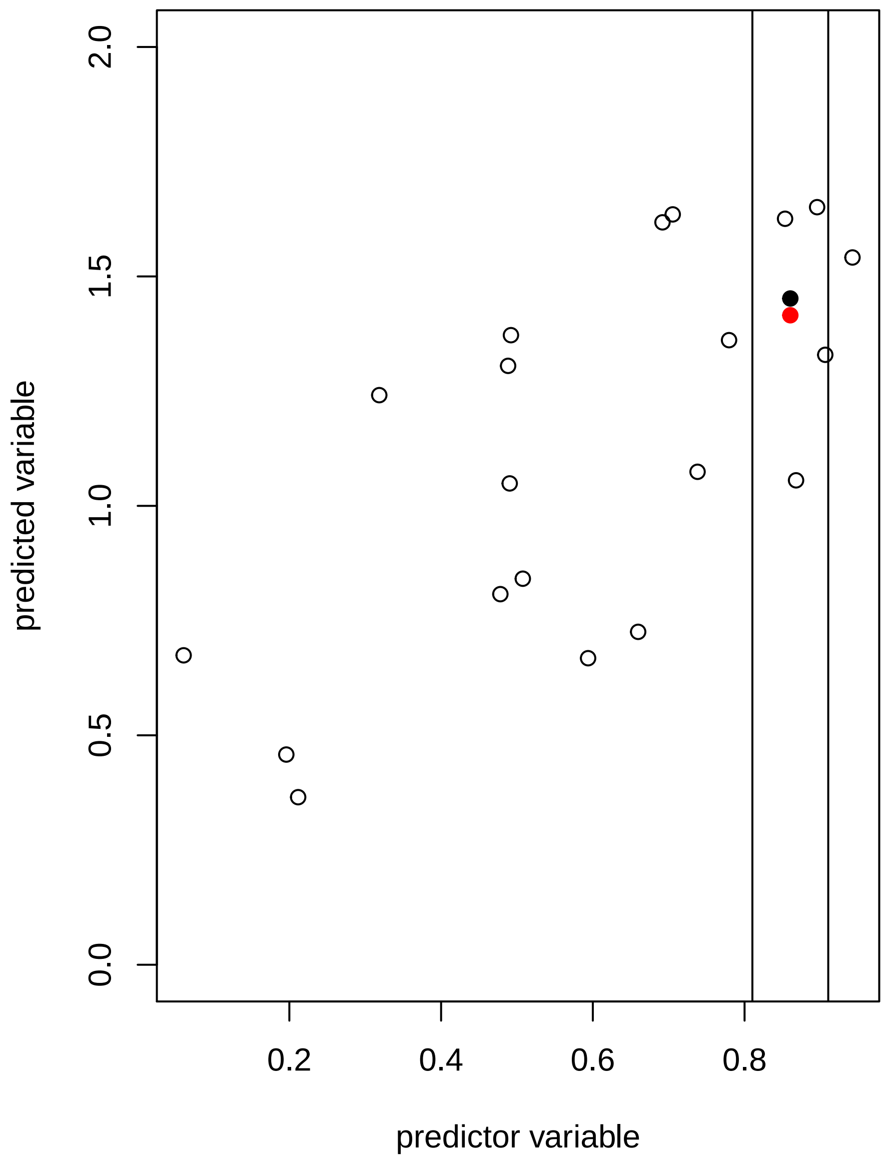
Deconstructing the model: K-Nearest Neighbor (kNN)

kNN is a nonparametric, almost stupidly simple method that just finds data points in the training set that are closest to each test case and uses them to make a prediction. kNN is asymptotically optimal as a predictor (https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm).

```r
set.seed(667)
# just using random samples here, with mild covariation
a = runif(20) # a predictor variable
b = a+runif(20) # a variable of interest to be predicted
par(mfrow=c(1,1))
plot(a,b,xlab='predictor variable',ylab='predicted variable',ylim=c(0,2))
# a new data point to predict
c=runif(1)
d=c+runif(1)
points(c,d,pch=19)
abline(v=c-.05)
abline(v=c+.05)
neighbs = which((a > c-.05) & (a<c+.05))
neighbs # these are the nearest neigbors (not doing a specific k here)
```

```
## [1]  6  8 12 18
```

```r
knn_est = mean(b[neighbs])  # make a prediction based on neighbs
points(c, knn_est, pch=19,col='red')  # plot it
```



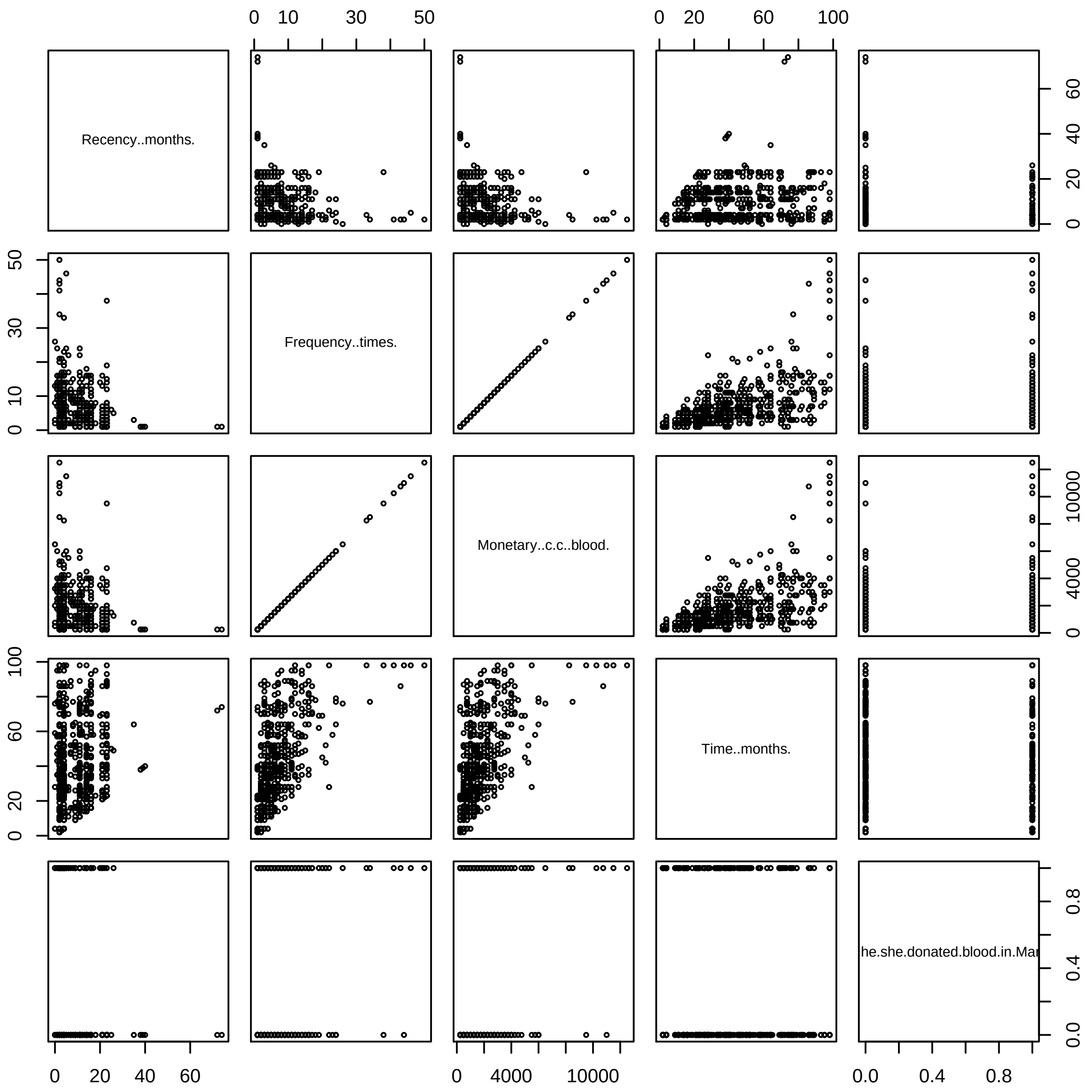
Predicting blood donation

Well, hopefully that was instructive. Now let's look at some actual data. This dataset is from a paper whose reference I have lost, trying to predict who will show up for blood drives, based on prior donation history.

```
trans = read.csv('transfusion.data',header=T) # read it in
head(trans)
```

```
##   Recency..months. Frequency..times. Monetary..c.c..blood. Time..months.
## 1                2                50                 12500            98
## 2                0                13                  3250            28
## 3                1                16                  4000            35
## 4                2                20                  5000            45
## 5                1                24                  6000            77
## 6                4                 4                  1000             4
##   whether.he.she.donated.blood.in.March.2007
## 1                                              1
## 2                                              1
## 3                                              1
## 4                                              1
## 5                                              0
## 6                                              0
```

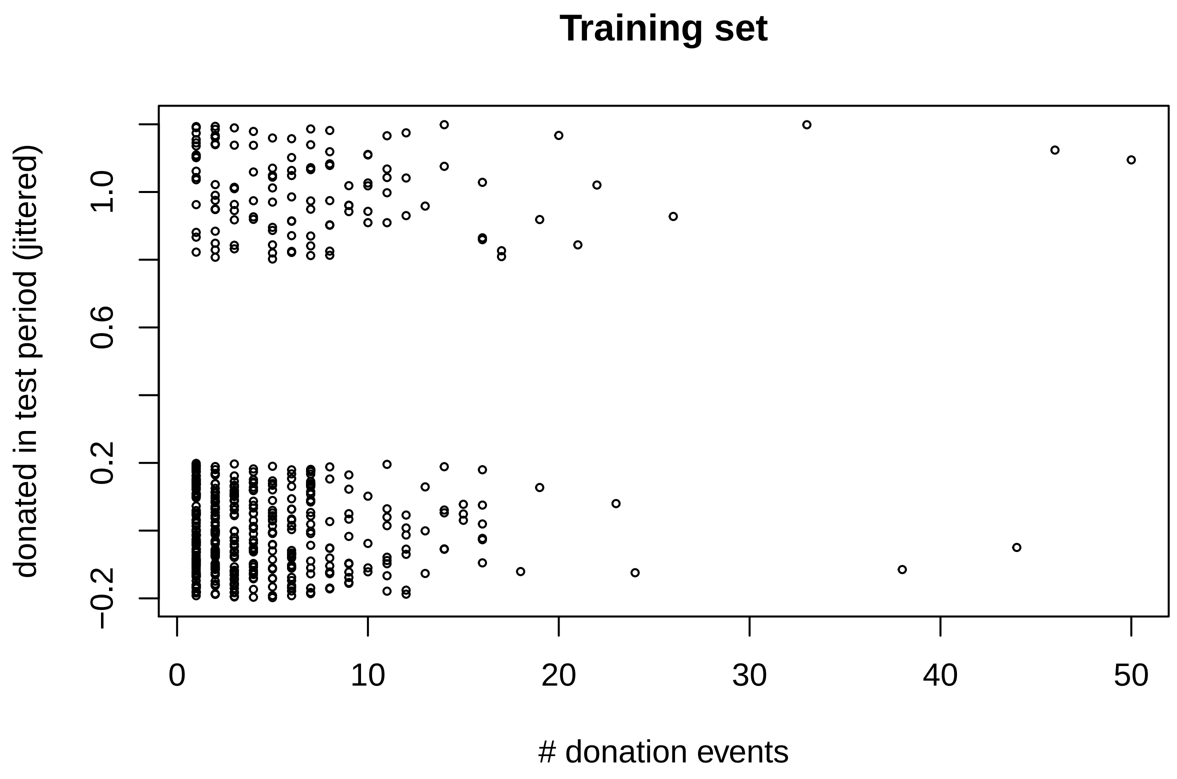
```r
plot(trans,cex=.5)
```



Obviously some of these things are more meaningful than other things. I will sorta naively fit the model based on everything, ignoring the possibility of interactions.

Using prediction to evaluate the model. I chose to randomly sample 500 observations to train, and test on the remaining 248.

```r
# training set
set.seed(666) # EXTREME
trainindex = sample(1:748,500)
train = trans[trainindex,]
# test set
test = trans[!(1:nrow(trans) %in% trainindex),]

# some utility functions
source('roc.R')

# plot one particular variable
plot(train$Frequency..times.,
    jitter(train$whether.he.she.donated.blood.in.March.2007),
    xlab='# donation events',ylab='donated in test period (jittered)',
    cex = .5 , main='Training set')
```



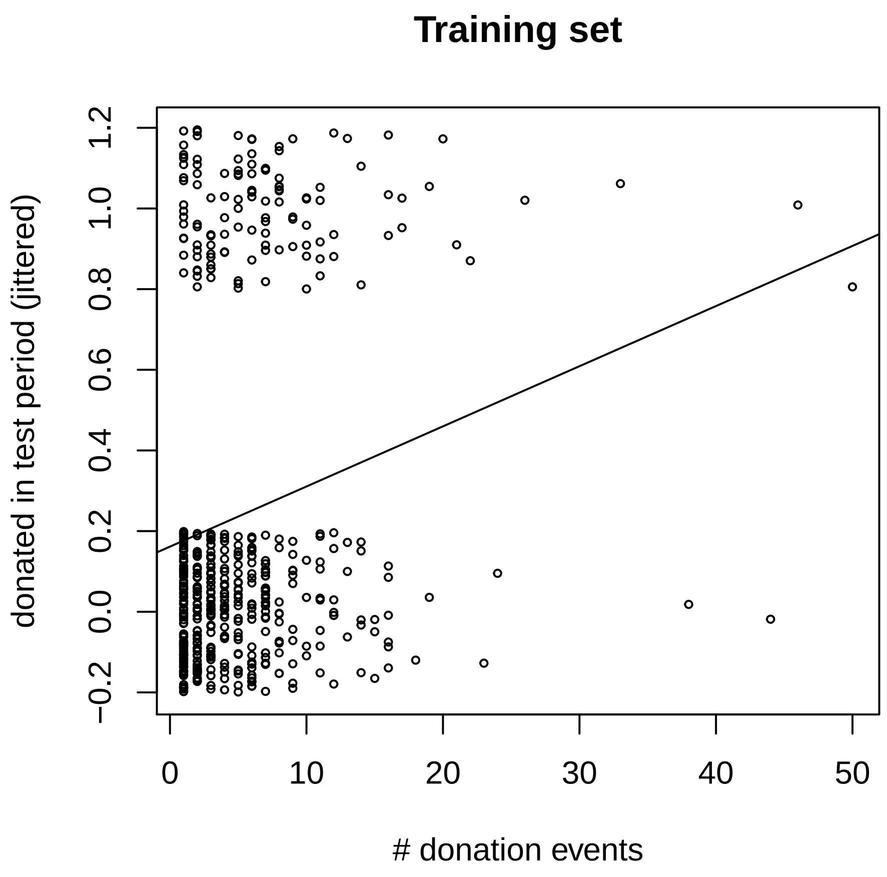
First, fit a linear model, which is ok but not very interesting.

```r
# fit the model
linmod = lm(whether.he.she.donated.blood.in.March.2007 ~
    Frequency..times., data = train)
str(linmod)
```

```
## List of 12
##  $ coefficients : Named num [1:2] 0.1615 0.0149
##   ..- attr(*, "names")= chr [1:2] "(Intercept)" "Frequency..times."
##  $ residuals    : Named num [1:500] -0.206 -0.191 -0.206 -0.191 -0.266 ...
##   ..- attr(*, "names")= chr [1:500] "580" "148" "730" "150" ...
##  $ effects      : Named num [1:500] -5.411 1.916 -0.194 -0.178 -0.259 ...
##   ..- attr(*, "names")= chr [1:500] "(Intercept)" "Frequency..times." "" "" ...
##  $ rank         : int 2
##  $ fitted.values: Named num [1:500] 0.206 0.191 0.206 0.191 0.266 ...
##   ..- attr(*, "names")= chr [1:500] "580" "148" "730" "150" ...
##  $ assign       : int [1:2] 0 1
##  $ qr           :List of 5
##   ..$ qr   : num [1:500, 1:2] -22.3607 0.0447 0.0447 0.0447 0.0447 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:500] "580" "148" "730" "150" ...
##   .. .. ..$ : chr [1:2] "(Intercept)" "Frequency..times."
##   .. ..- attr(*, "assign")= int [1:2] 0 1
##   ..$ qraux: num [1:2] 1.04 1.03
##   ..$ pivot: int [1:2] 1 2
##   ..$ tol  : num 1e-07
##   ..$ rank : int 2
##   ..- attr(*, "class")= chr "qr"
##  $ df.residual  : int 498
##  $ xlevels      : Named list()
##  $ call         : language lm(formula = whether.he.she.donated.blood.in.March.2007 ~ Frequency..time
##  $ terms        :Classes 'terms', 'formula' length 3 whether.he.she.donated.blood.in.March.2007 ~ Fr
##   .. ..- attr(*, "variables")= language list(whether.he.she.donated.blood.in.March.2007, Frequency..
##   .. ..- attr(*, "factors")= int [1:2, 1] 0 1
##   .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. ..$ : chr [1:2] "whether.he.she.donated.blood.in.March.2007" "Frequency..times."
##   .. .. .. ..$ : chr "Frequency..times."
##   .. ..- attr(*, "term.labels")= chr "Frequency..times."
##   .. ..- attr(*, "order")= int 1
##   .. ..- attr(*, "intercept")= int 1
##   .. ..- attr(*, "response")= int 1
##   .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. ..- attr(*, "predvars")= language list(whether.he.she.donated.blood.in.March.2007, Frequency..t
##   .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
##   .. .. ..- attr(*, "names")= chr [1:2] "whether.he.she.donated.blood.in.March.2007" "Frequency..tim
##  $ model        :'data.frame':   500 obs. of  2 variables:
##   ..$ whether.he.she.donated.blood.in.March.2007: int [1:500] 0 0 0 0 0 1 0 1 1 0 ...
##   ..$ Frequency..times.                         : int [1:500] 3 2 3 2 7 6 6 5 46 4 ...
##   ..- attr(*, "terms")=Classes 'terms', 'formula' length 3 whether.he.she.donated.blood.in.March.200
##   .. .. ..- attr(*, "variables")= language list(whether.he.she.donated.blood.in.March.2007, Frequenc
##   .. .. ..- attr(*, "factors")= int [1:2, 1] 0 1
##   .. .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. .. ..$ : chr [1:2] "whether.he.she.donated.blood.in.March.2007" "Frequency..times."
##   .. .. .. .. ..$ : chr "Frequency..times."
##   .. .. ..- attr(*, "term.labels")= chr "Frequency..times."
##   .. .. ..- attr(*, "order")= int 1
##   .. .. ..- attr(*, "intercept")= int 1
##   .. .. ..- attr(*, "response")= int 1
##   .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. .. ..- attr(*, "predvars")= language list(whether.he.she.donated.blood.in.March.2007, Frequency
##   .. .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
##   .. .. .. ..- attr(*, "names")= chr [1:2] "whether.he.she.donated.blood.in.March.2007" "Frequency..
##  - attr(*, "class")= chr "lm"
```

```
plot(train$Frequency..times.,
     jitter(train$whether.he.she.donated.blood.in.March.2007),
```

```r
    xlab='# donation events',ylab='donated in test period (jittered)',
    cex = .5 , main='Training set')

# things you can do with the fitted model object
abline(linmod)  # add the predicted function to the plot just generated
```



```r
# return various useful information about the model:
summary(linmod) # print a lot of results, in semi-human-readable table
```

```
##
## Call:
## lm(formula = whether.he.she.donated.blood.in.March.2007 ~ Frequency..times.,
##     data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8176 -0.2361 -0.1914 -0.1764  0.8236
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.161541   0.025798   6.262 8.23e-10 ***
## Frequency..times. 0.014911   0.003273   4.556 6.58e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.4205 on 498 degrees of freedom
## Multiple R-squared:  0.04001,    Adjusted R-squared:  0.03808
## F-statistic: 20.75 on 1 and 498 DF,  p-value: 6.577e-06
```

```
coef(linmod)    # coefficients (parameters)
```

```
##       (Intercept) Frequency..times.
##        0.16154059        0.01491094
```

```
confint(linmod) # confidence intervals
```

```
##                         2.5 %    97.5 %
## (Intercept)       0.110854123 0.2122271
## Frequency..times. 0.008480175 0.0213417
```

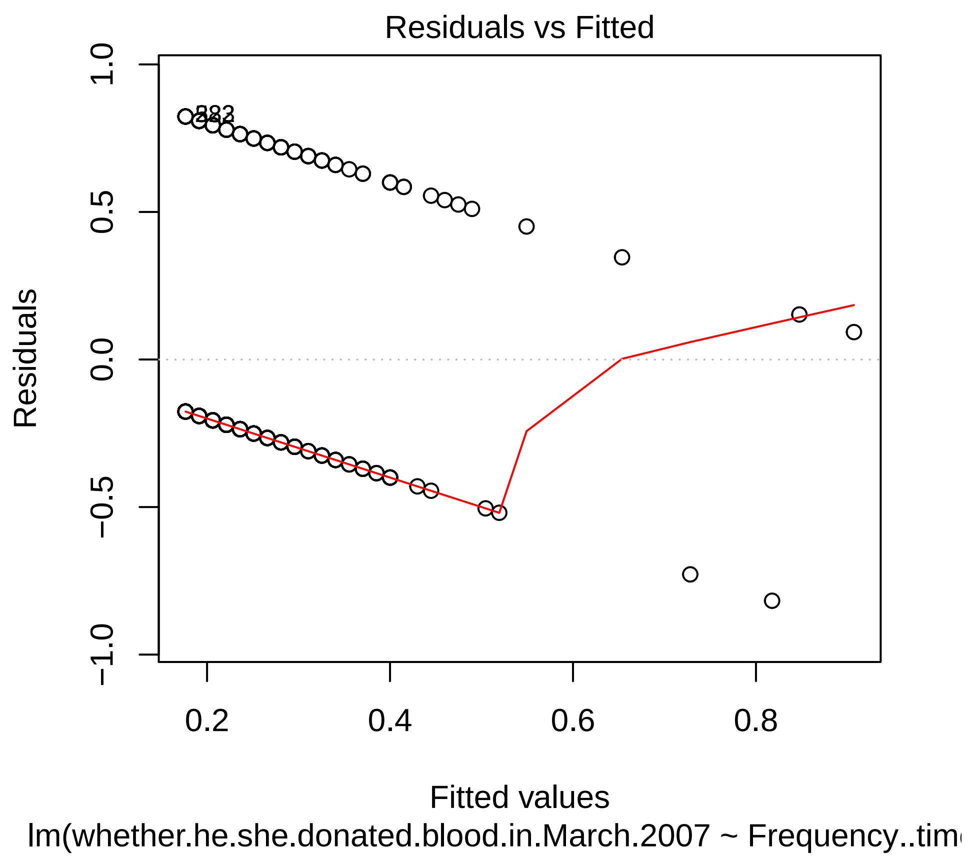
```
resid(linmod)[1:10] # residuals on the model -  printing out only first ten
```

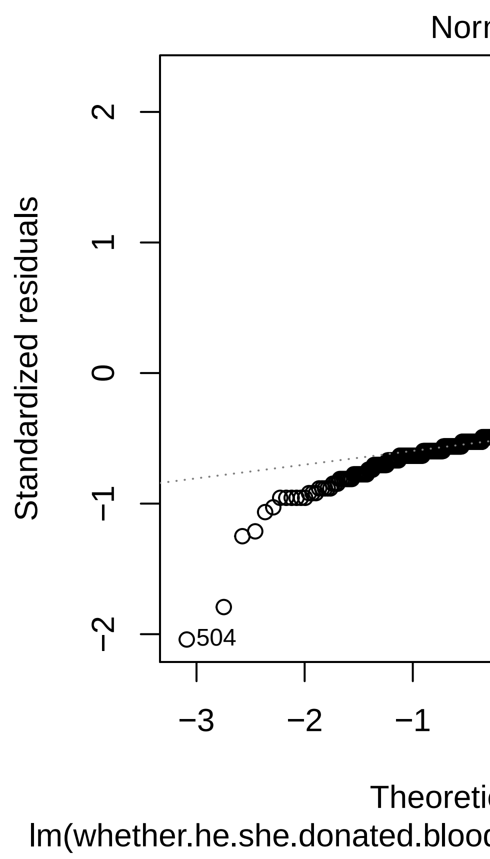
```
##        580        148        730        150        269        552
## -0.2062734 -0.1913625 -0.2062734 -0.1913625 -0.2659171  0.7489938
##        727        370         10        193
## -0.2510062  0.7639047  0.1525564 -0.2211843
```

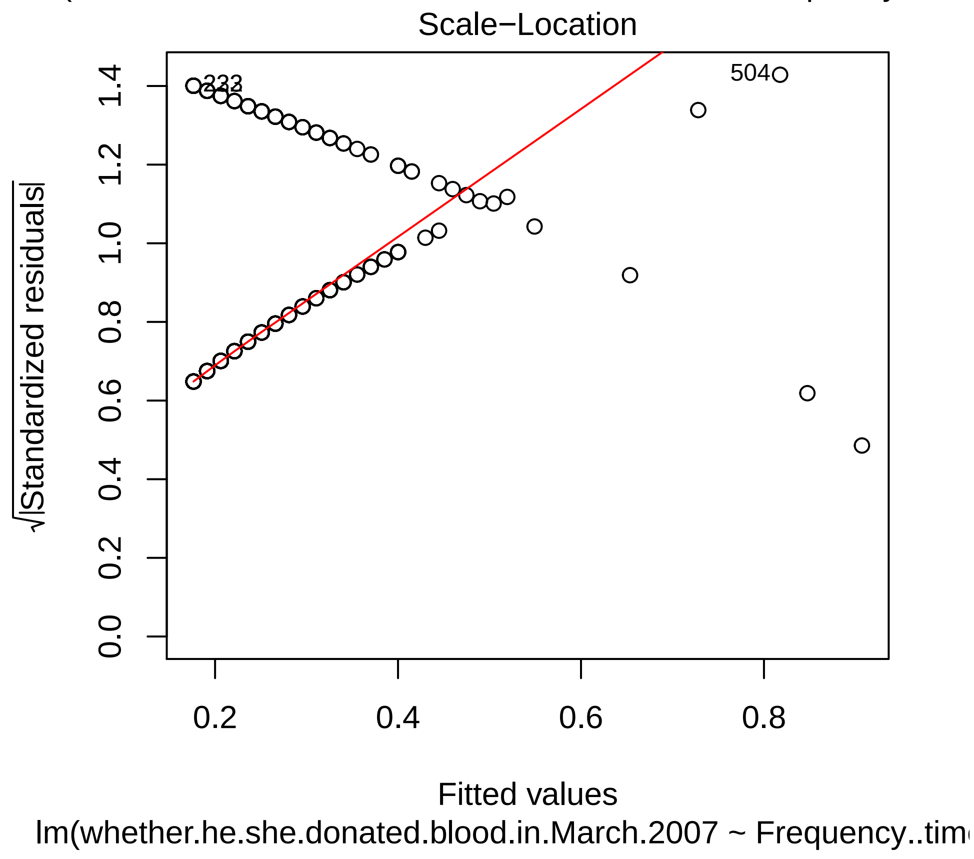
```
anova(linmod)   # anova table
```

```
## Analysis of Variance Table
##
## Response: whether.he.she.donated.blood.in.March.2007
##                    Df Sum Sq Mean Sq F value    Pr(>F)
## Frequency..times.   1  3.669  3.6693  20.754 6.577e-06 ***
## Residuals         498 88.049  0.1768
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# this would plot lots of model fit info, which may or may not be useful:
plot(linmod)
```

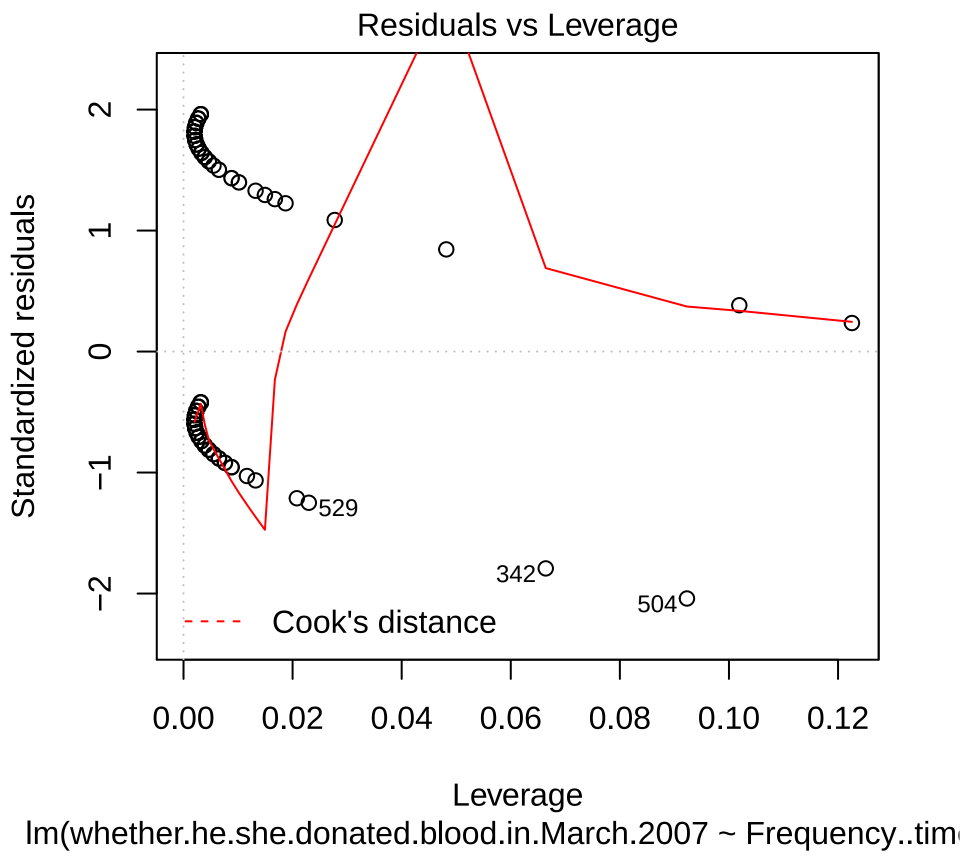

Residuals vs Fitted
Residuals
1.0
0.5
0.0
−0.5
−1.0
0.2
0.4
0.6
0.8
Fitted values
lm(whether.he.she.donated.blood.in.March.2007 ~ Frequency..tim
582


Norr
Standardized residuals
2
1
0
−1
−2
−3
−2
−1
504
Theoretic
lm(whether.he.she.donated.blood

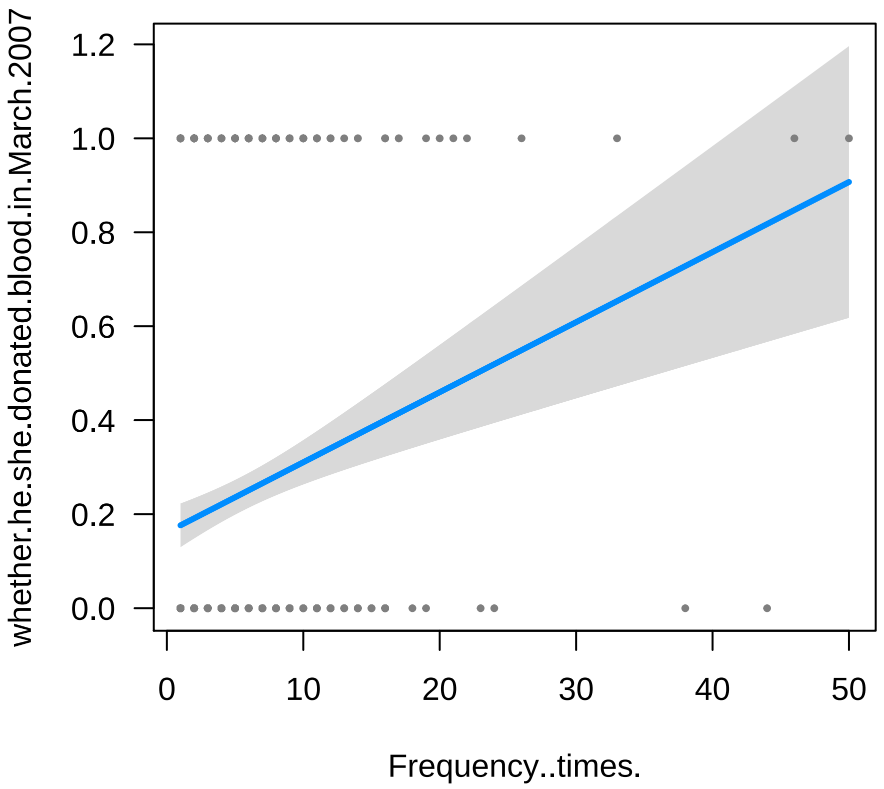

Scale−Location
√|Standardized residuals|
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0.2
0.4
0.6
0.8
Fitted values
lm(whether.he.she.donated.blood.in.March.2007 ~ Frequency..tim
232
504

```r
# alternate visualization method
require(visreg)
```

```
## Loading required package: visreg
```



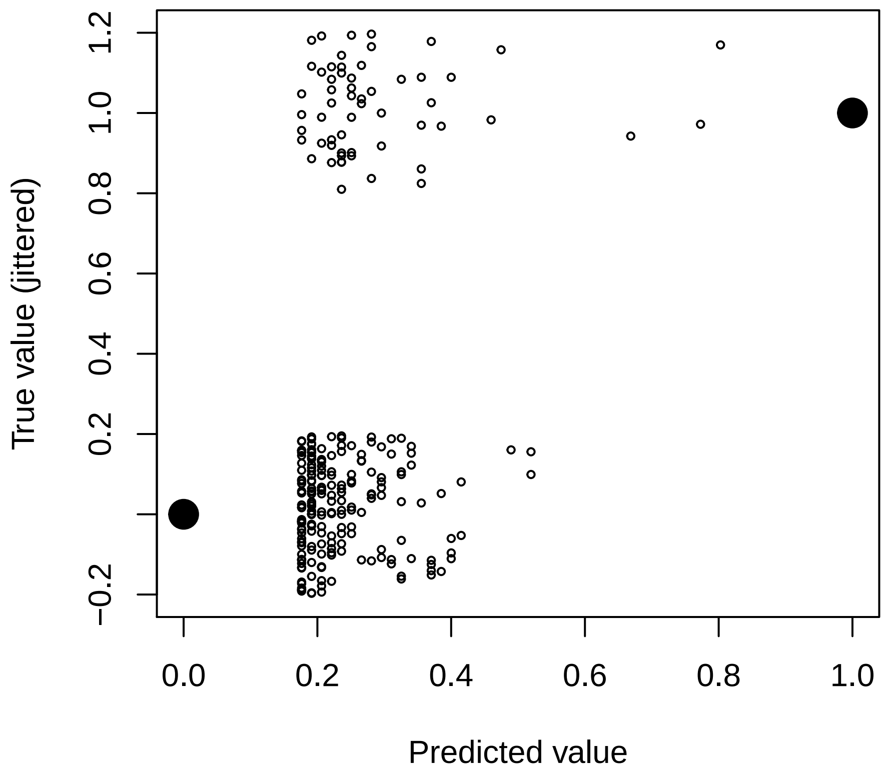
```r
visreg(linmod)
```



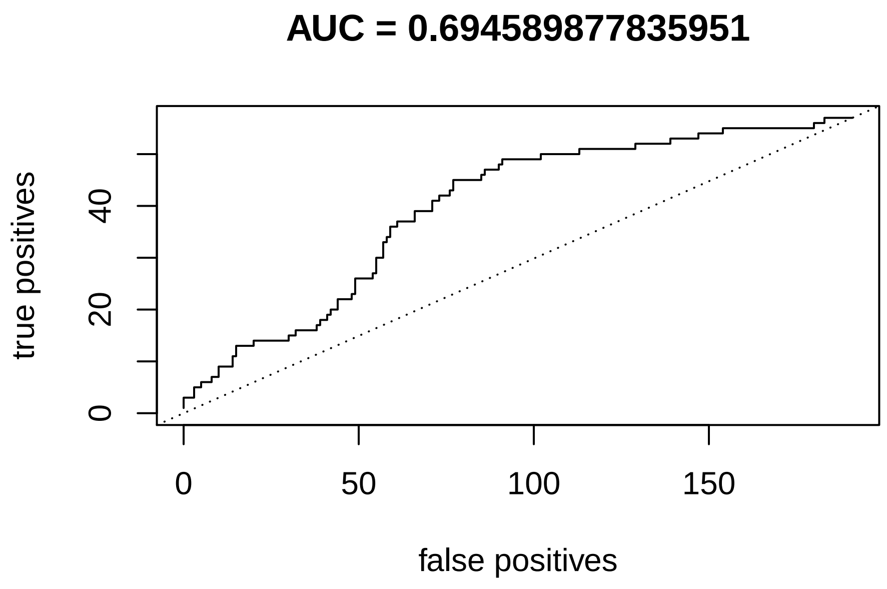
So we had a low p-value, which is good right? Problem solved, everyone go home.

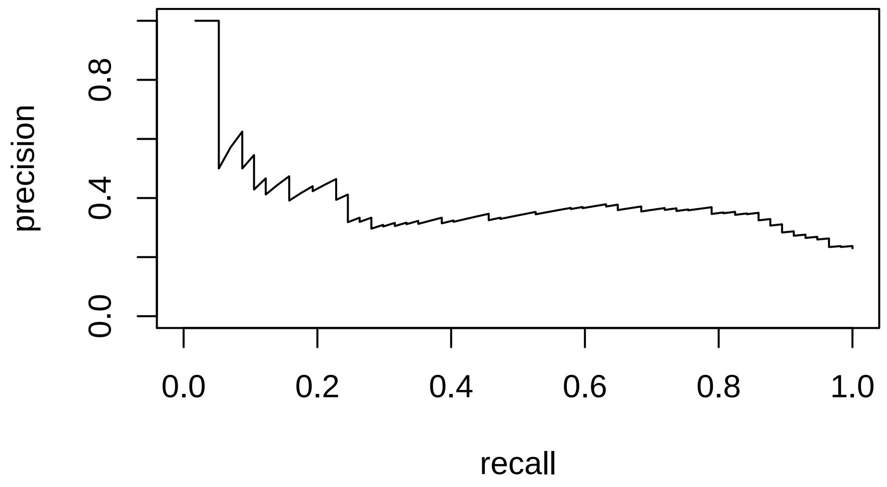
Except this is obviously a really crappy model. This can be shown if we try to predict test values (new data that wasn't used to build the model, just plugging new values into the model function) and compare them to the actual values of the test outcome.

```
linpred = predict(linmod,newdata=test)
linpredplot = plot(
    linpred,
    jitter(test$whether.he.she.donated.blood.in.March.2007),
    ylab='True value (jittered)', xlab='Predicted value',
    ylim = c(-.2,1.2), xlim = c(0,1), cex = .5)

points( c(0,1), c(0,1), cex = 2, pch = 19 )
```



```r
prediction = cbind(linpred,test[,5])
a = ROC(prediction)
```





Not great. There are also some numerical summaries of model fit that various people use (besides R^2).

```
# Akaike information criterion:
# 2(num parameters) - 2ln( L(model) ) [lower is better!!!]
AIC(linmod)
```

```
## [1] 556.5793
```

```
# stolen from https://www.kaggle.com/c/bioresponse/forums/t/1576/r-code-for-logloss
# prettied up from that to make more readable
LogLoss = function(actual, predicted)   {
# for explanation see https://en.wikipedia.org/wiki/Loss_function
    result = -1/length(actual) *
    sum(
    actual * log(predicted) + # true prediction failures
    (1-actual) * log(1-predicted) # false prediction failures
    )
    return(result)  }

# note that this makes use of training set
LogLoss( test$whether.he.she.donated.blood.in.March.2007, linpred )
```

```
## [1] 0.5098918
```

```
# AUC from the ROC curve above also is such a measure.
# you can even use a U-test to sort of evaluate the quality of the predictions:
wilcox.test(
    linpred[test$whether.he.she.donated.blood.in.March.2007 == 1],
    linpred[test$whether.he.she.donated.blood.in.March.2007 == 0]
    )
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  linpred[test$whether.he.she.donated.blood.in.March.2007 == 1] and linpred[test$whether.he.she
## W = 7442.5, p-value = 2.273e-05
## alternative hypothesis: true location shift is not equal to 0
```

```
# or even something as naive as correlations
cor(test$whether.he.she.donated.blood.in.March.2007, linpred,method='spearman')
```

```
## [1] 0.2696155
```

can in principle make the model more complicated by adding more variables, and compare to the old model.

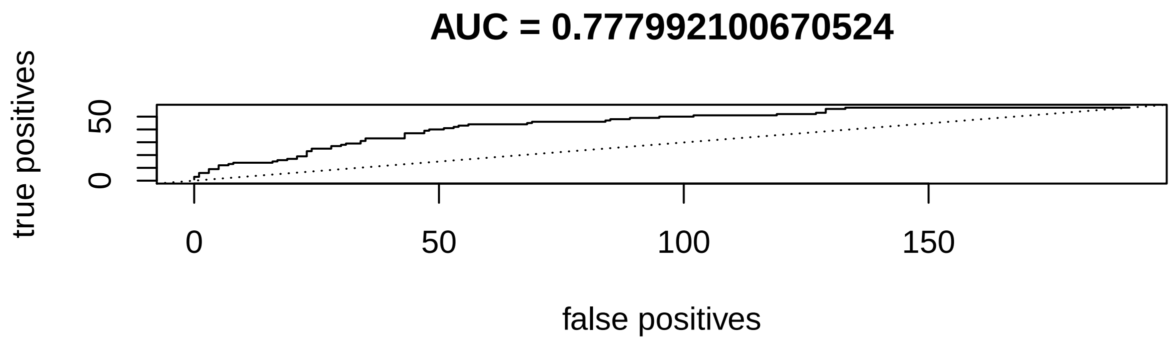
```
multilinmod = lm(whether.he.she.donated.blood.in.March.2007 ~
    Frequency..times. + Recency..months.,data = train)

# model comparisons!!
AIC(multilinmod,linmod)
```

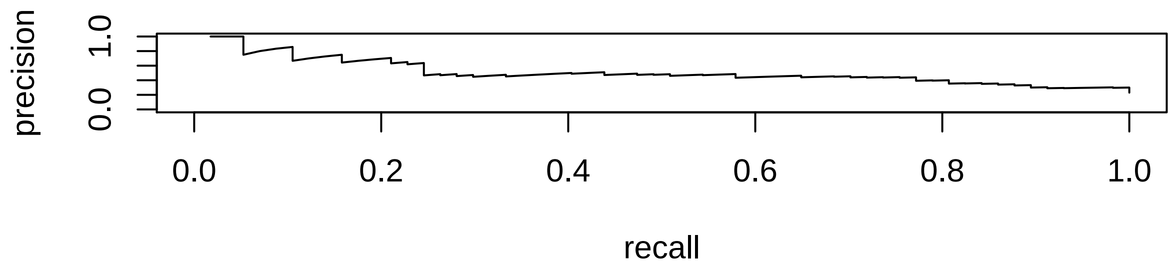
```
##             df      AIC
## multilinmod  4 534.0743
## linmod       3 556.5793
```

```
anova(multilinmod,linmod)
```

```
## Analysis of Variance Table
##
## Model 1: whether.he.she.donated.blood.in.March.2007 ~ Frequency..times. +
##     Recency..months.
## Model 2: whether.he.she.donated.blood.in.March.2007 ~ Frequency..times.
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    497 83.837
## 2    498 88.049 -1   -4.2112 24.965 8.102e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
multipredict = cbind(predict(multilinmod,newdata=test),test[,5])
a = ROC(multipredict)
```





```
c(logLik(multilinmod),logLik(linmod))
```

```
## [1] -263.0372 -275.2896
```

```
c(LogLoss(multipredict[,2],multipredict[,1]),LogLoss(multipredict[,2],linpred))
```

```
## Warning in log(predicted): NaNs produced
```

```
## [1]       NaN 0.5098918
```

Try instead a logistic regression: a generalized linear model (GLM) of the family “binomial”. That is, it expects the outcome variable (blood donation) to be distributed as a binomial (0/1) random variable. The predictor “generalizes” a linear fit using the logistic function to be able to make discrete 0/1 predictions.

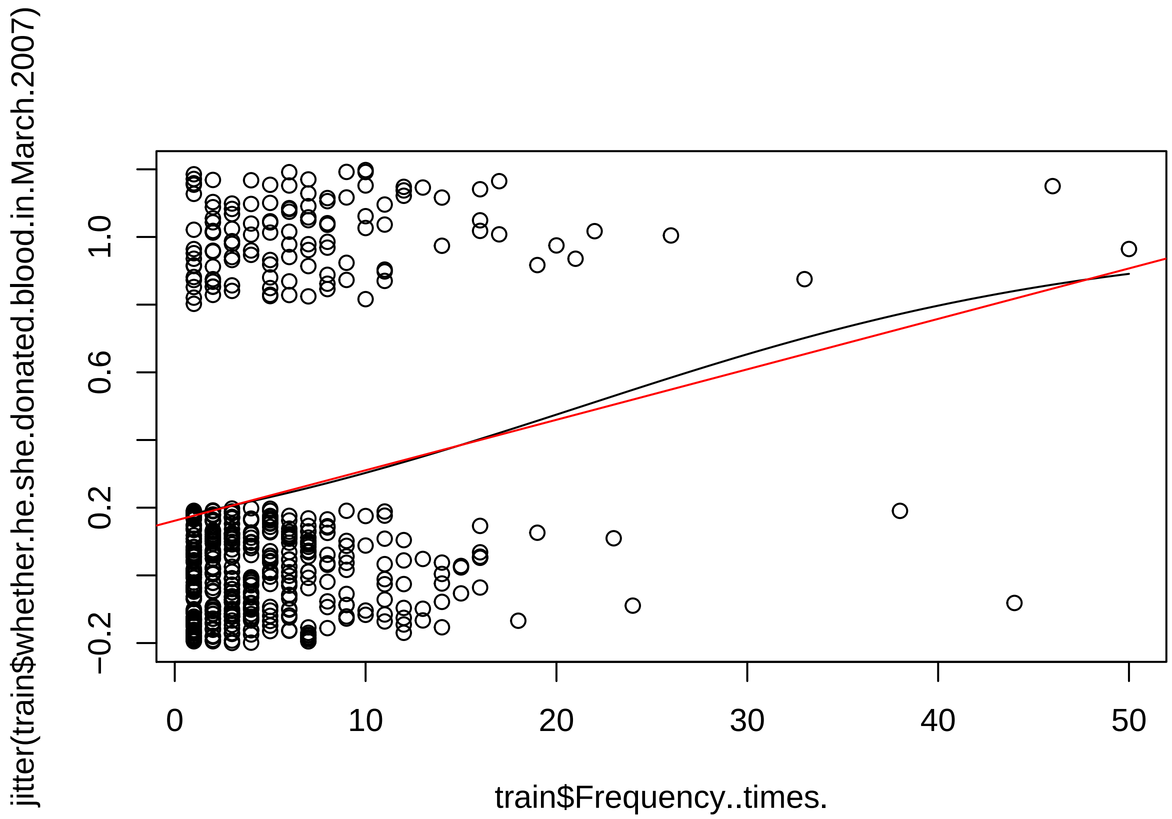
```
trainfit = glm(whether.he.she.donated.blood.in.March.2007 ~
    Frequency..times.,family='binomial',data=train )

class(linmod)
```

```
## [1] "lm"
```

```
class(trainfit)
```

```
## [1] "glm" "lm"
```

```
# plot out predictions of 2 models
par(mfrow = c(1,1) )
plot(train$Frequency..times.,jitter(train$whether.he.she.donated.blood.in.March.2007))
curve( predict( trainfit, data.frame(Frequency..times.=x), type='response' ), add=TRUE )
abline(linmod,col='red')
```



```
AIC(trainfit,linmod,multilinmod)
```

```
##             df      AIC
## trainfit     2 539.5490
## linmod       3 556.5793
## multilinmod  4 534.0743
```

```
# add EVERYTHING to the model
trainfit = glm(whether.he.she.donated.blood.in.March.2007 ~ Recency..months. *
     Frequency..times. * Monetary..c.c..blood.
    * Time..months.,family='binomial',data=train
    )

# summarize it a little...
AIC(trainfit,multilinmod,linmod)
```

```
##             df      AIC
## trainfit    12 503.4156
## multilinmod  4 534.0743
## linmod       3 556.5793
```

```r
summary(trainfit)
```

```
##
## Call:
## glm(formula = whether.he.she.donated.blood.in.March.2007 ~ Recency..months. *
##     Frequency..times. * Monetary..c.c..blood. * Time..months.,
##     family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7693  -0.7128  -0.5314  -0.2256   2.3888
##
## Coefficients: (4 not defined because of singularities)
##                                                                     Estimate
## (Intercept)                                                       -1.539e+00
## Recency..months.                                                  -9.733e-03
## Frequency..times.                                                  5.587e-01
## Monetary..c.c..blood.                                                     NA
## Time..months.                                                     -2.571e-02
## Recency..months.:Frequency..times.                                -2.159e-02
## Recency..months.:Monetary..c.c..blood.                                    NA
## Frequency..times.:Monetary..c.c..blood.                           -4.609e-05
## Recency..months.:Time..months.                                    -5.562e-04
## Frequency..times.:Time..months.                                   -4.042e-03
## Monetary..c.c..blood.:Time..months.                                       NA
## Recency..months.:Frequency..times.:Monetary..c.c..blood.          -2.729e-06
## Recency..months.:Frequency..times.:Time..months.                   3.940e-04
## Recency..months.:Monetary..c.c..blood.:Time..months.                      NA
## Frequency..times.:Monetary..c.c..blood.:Time..months.              4.454e-07
## Recency..months.:Frequency..times.:Monetary..c.c..blood.:Time..months.  7.231e-09
##                                                                    Std. Error
## (Intercept)                                                         4.781e-01
## Recency..months.                                                    6.094e-02
## Frequency..times.                                                   1.999e-01
## Monetary..c.c..blood.                                                      NA
## Time..months.                                                       1.818e-02
## Recency..months.:Frequency..times.                                  2.585e-02
## Recency..months.:Monetary..c.c..blood.                                     NA
## Frequency..times.:Monetary..c.c..blood.                             5.760e-05
## Recency..months.:Time..months.                                      1.961e-03
## Frequency..times.:Time..months.                                     2.364e-03
## Monetary..c.c..blood.:Time..months.                                        NA
## Recency..months.:Frequency..times.:Monetary..c.c..blood.            8.702e-06
## Recency..months.:Frequency..times.:Time..months.                    3.235e-04
## Recency..months.:Monetary..c.c..blood.:Time..months.                       NA
## Frequency..times.:Monetary..c.c..blood.:Time..months.               5.273e-07
## Recency..months.:Frequency..times.:Monetary..c.c..blood.:Time..months.  8.139e-08
##                                                                    z value
## (Intercept)                                                         -3.218
## Recency..months.                                                        -0.160
## Frequency..times.                                                        2.795
## Monetary..c.c..blood.                                                       NA
## Time..months.                                                           -1.414
## Recency..months.:Frequency..times.                                      -0.835
## Recency..months.:Monetary..c.c..blood.                                      NA
## Frequency..times.:Monetary..c.c..blood.                                 -0.800
## Recency..months.:Time..months.                                          -0.284
## Frequency..times.:Time..months.                                         -1.710
## Monetary..c.c..blood.:Time..months.                                         NA
## Recency..months.:Frequency..times.:Monetary..c.c..blood.                -0.314
## Recency..months.:Frequency..times.:Time..months.                         1.218
## Recency..months.:Monetary..c.c..blood.:Time..months.                        NA
## Frequency..times.:Monetary..c.c..blood.:Time..months.                    0.845
## Recency..months.:Frequency..times.:Monetary..c.c..blood.:Time..months.   0.089
##                                                                        Pr(>|z|)
## (Intercept)                                                             0.00129
## Recency..months.                                                        0.87311
## Frequency..times.                                                       0.00519
## Monetary..c.c..blood.                                                        NA
## Time..months.                                                           0.15732
## Recency..months.:Frequency..times.                                      0.40357
## Recency..months.:Monetary..c.c..blood.                                       NA
## Frequency..times.:Monetary..c.c..blood.                                 0.42369
## Recency..months.:Time..months.                                          0.77674
## Frequency..times.:Time..months.                                         0.08725
## Monetary..c.c..blood.:Time..months.                                          NA
## Recency..months.:Frequency..times.:Monetary..c.c..blood.                0.75387
## Recency..months.:Frequency..times.:Time..months.                        0.22324
## Recency..months.:Monetary..c.c..blood.:Time..months.                         NA
## Frequency..times.:Monetary..c.c..blood.:Time..months.                   0.39822
## Recency..months.:Frequency..times.:Monetary..c.c..blood.:Time..months.  0.92921
##
## (Intercept)                                                            **
## Recency..months.
## Frequency..times.                                                      **
## Monetary..c.c..blood.
## Time..months.
## Recency..months.:Frequency..times.
## Recency..months.:Monetary..c.c..blood.
## Frequency..times.:Monetary..c.c..blood.
## Recency..months.:Time..months.
## Frequency..times.:Time..months.                                        .
## Monetary..c.c..blood.:Time..months.
## Recency..months.:Frequency..times.:Monetary..c.c..blood.
## Recency..months.:Frequency..times.:Time..months.
## Recency..months.:Monetary..c.c..blood.:Time..months.
## Frequency..times.:Monetary..c.c..blood.:Time..months.
## Recency..months.:Frequency..times.:Monetary..c.c..blood.:Time..months.
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 553.37  on 499  degrees of freedom
## Residual deviance: 479.42  on 488  degrees of freedom
## AIC: 503.42
##
## Number of Fisher Scoring iterations: 6
```

```r
# automated model selection
stepped = step(trainfit)
```

```
## Start:  AIC=503.42
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. *
##     Frequency..times. * Monetary..c.c..blood. * Time..months.
##
##                                                                         Df
## - Recency..months.:Frequency..times.:Monetary..c.c..blood.:Time..months.  1
## <none>
##                                                                         Deviance
## - Recency..months.:Frequency..times.:Monetary..c.c..blood.:Time..months.   479.42
## <none>                                                                     479.42
##                                                                            AIC
## - Recency..months.:Frequency..times.:Monetary..c.c..blood.:Time..months. 501.42
## <none>                                                                   503.42
##
## Step:  AIC=501.42
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Recency..months.:Frequency..times. + Recency..months.:Monetary..c.c..blood. +
##     Frequency..times.:Monetary..c.c..blood. + Recency..months.:Time..months. +
##     Frequency..times.:Time..months. + Monetary..c.c..blood.:Time..months. +
##     Recency..months.:Frequency..times.:Monetary..c.c..blood. +
##     Recency..months.:Frequency..times.:Time..months. + Recency..months.:Monetary..c.c..blood.:Time..
##     Frequency..times.:Monetary..c.c..blood.:Time..months.
##
##
## Step:  AIC=501.42
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Recency..months.:Frequency..times. + Recency..months.:Monetary..c.c..blood. +
##     Frequency..times.:Monetary..c.c..blood. + Recency..months.:Time..months. +
##     Frequency..times.:Time..months. + Monetary..c.c..blood.:Time..months. +
##     Recency..months.:Frequency..times.:Monetary..c.c..blood. +
##     Recency..months.:Frequency..times.:Time..months. + Frequency..times.:Monetary..c.c..blood.:Time.
##
##                                                          Df Deviance
## - Recency..months.:Frequency..times.:Monetary..c.c..blood.  1   480.52
## <none>                                                         479.42
## - Frequency..times.:Monetary..c.c..blood.:Time..months.     1   481.68
## - Recency..months.:Frequency..times.:Time..months.          1   481.88
##                                                               AIC
## - Recency..months.:Frequency..times.:Monetary..c.c..blood. 500.52
## <none>                                                     501.42
## - Frequency..times.:Monetary..c.c..blood.:Time..months.    501.68
## - Recency..months.:Frequency..times.:Time..months.         501.88
##
```

```
## Step:  AIC=500.52
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Recency..months.:Frequency..times. + Recency..months.:Monetary..c.c..blood. +
##     Frequency..times.:Monetary..c.c..blood. + Recency..months.:Time..months. +
##     Frequency..times.:Time..months. + Monetary..c.c..blood.:Time..months. +
##     Recency..months.:Frequency..times.:Time..months. + Frequency..times.:Monetary..c.c..blood.:Time.
##
##
## Step:  AIC=500.52
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Recency..months.:Frequency..times. + Frequency..times.:Monetary..c.c..blood. +
##     Recency..months.:Time..months. + Frequency..times.:Time..months. +
##     Monetary..c.c..blood.:Time..months. + Recency..months.:Frequency..times.:Time..months. +
##     Frequency..times.:Monetary..c.c..blood.:Time..months.
##
##                                                    Df Deviance    AIC
## - Recency..months.:Frequency..times.:Time..months.       1   482.16 500.16
## <none>                                                 480.52 500.52
## - Frequency..times.:Monetary..c.c..blood.:Time..months.  1   482.92 500.92
##
## Step:  AIC=500.16
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Recency..months.:Frequency..times. + Frequency..times.:Monetary..c.c..blood. +
##     Recency..months.:Time..months. + Frequency..times.:Time..months. +
##     Monetary..c.c..blood.:Time..months. + Frequency..times.:Monetary..c.c..blood.:Time..months.
##
##                                                    Df Deviance    AIC
## - Recency..months.:Time..months.                         1   483.21 499.21
## <none>                                                 482.16 500.16
## - Frequency..times.:Monetary..c.c..blood.:Time..months.  1   484.25 500.25
## - Recency..months.:Frequency..times.                     1   484.80 500.80
##
## Step:  AIC=499.21
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Recency..months.:Frequency..times. + Frequency..times.:Monetary..c.c..blood. +
##     Frequency..times.:Time..months. + Monetary..c.c..blood.:Time..months. +
##     Frequency..times.:Monetary..c.c..blood.:Time..months.
##
##                                                    Df Deviance    AIC
## - Recency..months.:Frequency..times.                     1   484.84 498.84
## - Frequency..times.:Monetary..c.c..blood.:Time..months.  1   485.17 499.17
## <none>                                                 483.21 499.21
##
## Step:  AIC=498.84
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Frequency..times.:Monetary..c.c..blood. + Frequency..times.:Time..months. +
##     Monetary..c.c..blood.:Time..months. + Frequency..times.:Monetary..c.c..blood.:Time..months.
##
##                                                    Df Deviance    AIC
## - Frequency..times.:Monetary..c.c..blood.:Time..months.  1   486.43 498.43
## <none>                                                        484.84 498.84
## - Recency..months.                                        1   499.05 511.05
##
## Step:  AIC=498.43
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Frequency..times.:Monetary..c.c..blood. + Frequency..times.:Time..months. +
##     Monetary..c.c..blood.:Time..months.
##
##
## Step:  AIC=498.43
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Frequency..times.:Monetary..c.c..blood. + Frequency..times.:Time..months.
##
##                                          Df Deviance    AIC
## - Frequency..times.:Monetary..c.c..blood.  1   486.66 496.66
## - Frequency..times.:Time..months.          1   487.31 497.31
## <none>                                         486.43 498.43
## - Recency..months.                         1   501.93 511.93
##
## Step:  AIC=496.66
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Monetary..c.c..blood. + Time..months. +
##     Frequency..times.:Time..months.
##
##
## Step:  AIC=496.66
## whether.he.she.donated.blood.in.March.2007 ~ Recency..months. +
##     Frequency..times. + Time..months. + Frequency..times.:Time..months.
##
##                                  Df Deviance    AIC
## <none>                                 486.66 496.66
## - Frequency..times.:Time..months.  1   493.64 501.64
## - Recency..months.                 1   503.61 511.61
```

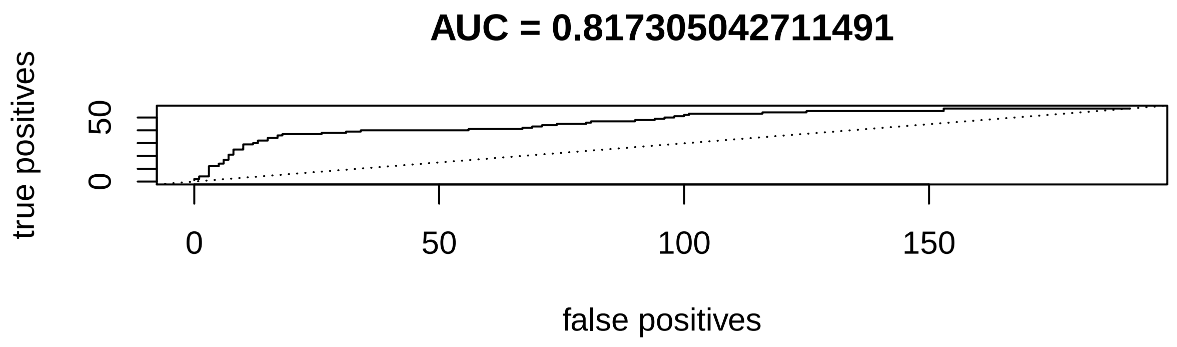
```
# do some predictions
predictor = predict.glm(trainfit,newdata=test)
```

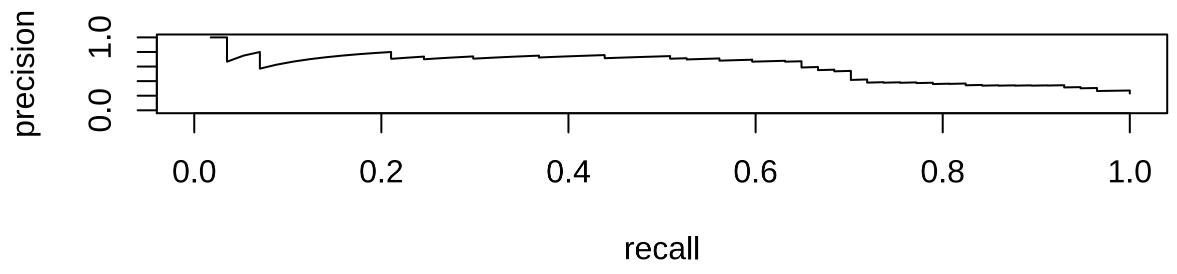
```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type =
## ifelse(type == : prediction from a rank-deficient fit may be misleading
```

```
prediction = cbind(predictor,test[,5])
colnames(prediction) = c( 'prediction','true values')

#curve( predict( trainfit, data.frame(Frequency..times.=x), type='response' ), add=TRUE, col='blue')

# various looks at prediction success
cor(prediction[,1],prediction[,2],method='spearman')
```

```
## [1] 0.4621751
```

```
#LogLoss(prediction[,1],prediction[,2])

a=ROC(prediction)
```

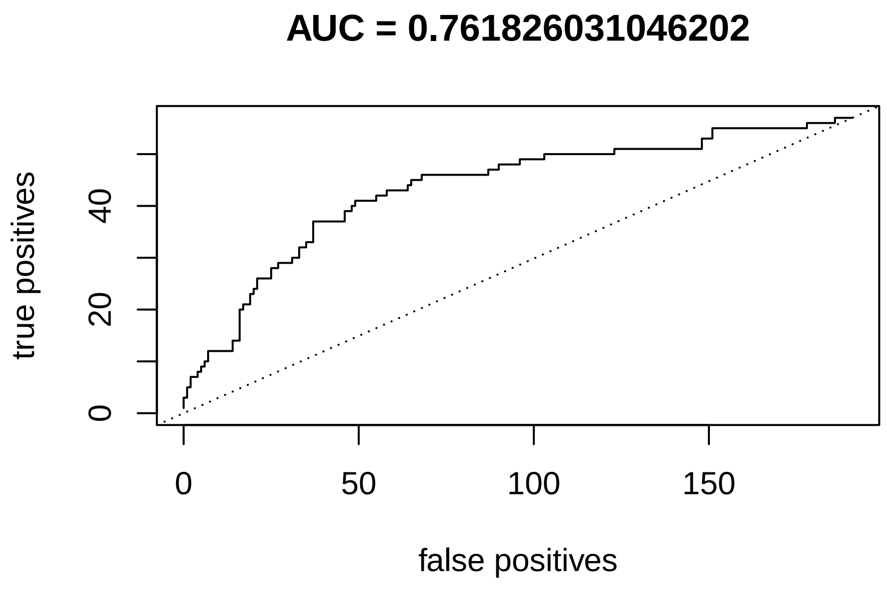


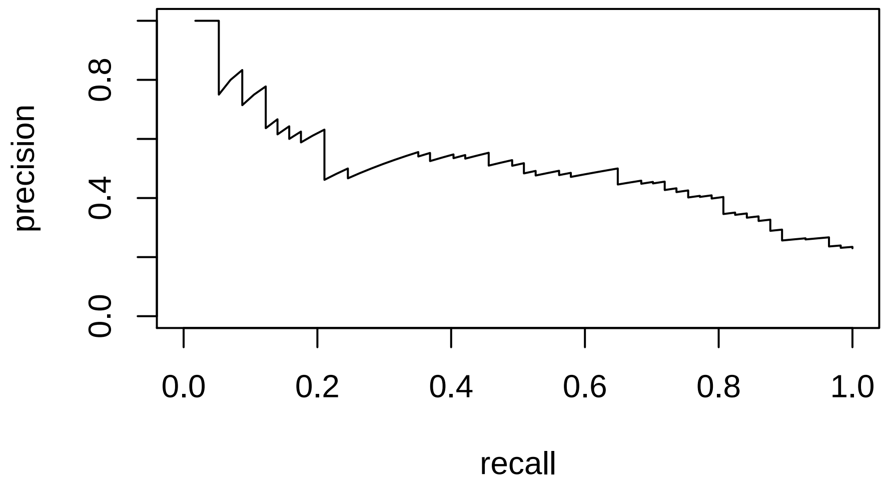


So it's okay (i.e. AUC>0.8).

```
curated_fit = glm(whether.he.she.donated.blood.in.March.2007 ~
    Frequency..times.
    + Time..months.,family='binomial',
    data=train)
curated_prediction = predict.glm(curated_fit,newdata=test)

prediction = cbind(curated_prediction,test[,5])
```

```
a=ROC(prediction)
```





Didn't really change much (Figure 3). Lost a little AUC, but not much for removing 2 explanatory variables in slavish devotion to occam's razor. Precision seems to fall apart a bit, though. While logistic regression is nice and simple, it is not doing a super job, so I will move on to see if anything else does better.

Naive Bayes

Naive Bayes is an attractively simple classification technique. It is similar to the initial logistic regression implemented above, because of its assumption of independence of predictor variables. It uses a straightforward interpretation of Bayes' rule to compute probabilities of each variable belonging to each class. While we only have a binary outcome, it is possible that NB will perform better for some reason.

```r
require(e1071)
```

```
## Loading required package: e1071
## Warning: package 'e1071' was built under R version 3.2.2
```

```
# this function wants response to be a factor
classifier = naiveBayes(train[,1:4],as.factor(train[,5]))
class(classifier)
```

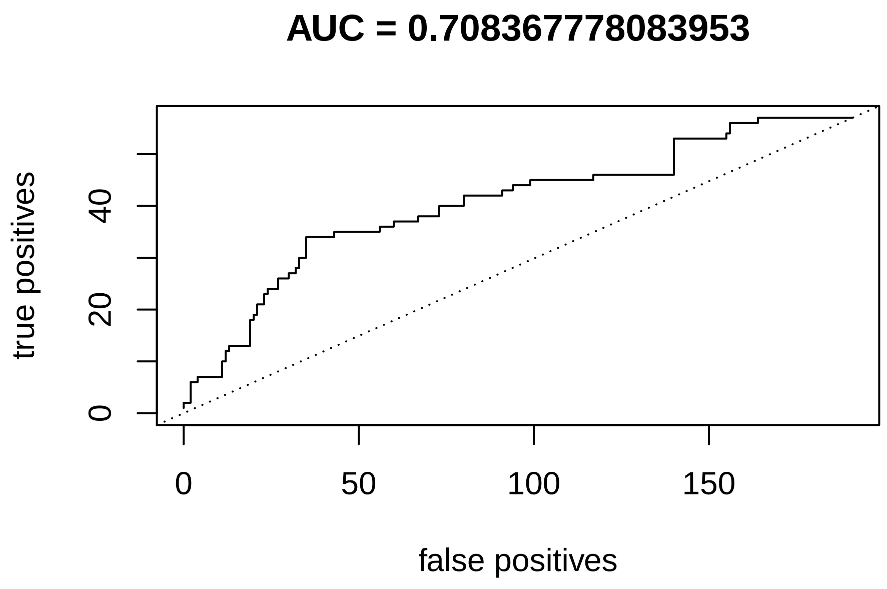
```
## [1] "naiveBayes"
```

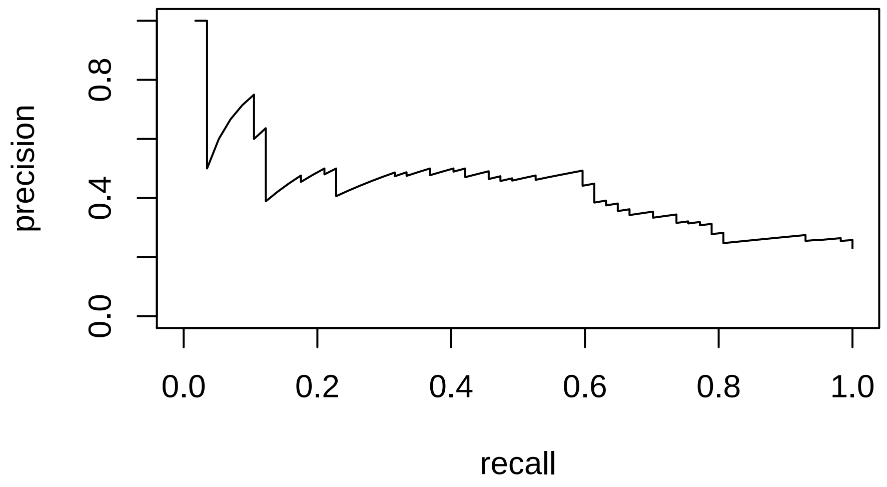
```
str(classifier)
```

```
## List of 4
##  $ apriori: 'table' int [1:2(1d)] 379 121
##   ..- attr(*, "dimnames")=List of 1
##   .. ..$ as.factor(train[, 5]): chr [1:2] "0" "1"
##  $ tables :List of 4
##   ..$ Recency..months.      : num [1:2, 1:2] 10.84 6.07 8.79 5.59
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ as.factor(train[, 5]): chr [1:2] "0" "1"
##   .. .. ..$ Recency..months.      : NULL
##   ..$ Frequency..times.     : num [1:2, 1:2] 4.75 7.43 4.79 7.73
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ as.factor(train[, 5]): chr [1:2] "0" "1"
##   .. .. ..$ Frequency..times.     : NULL
##   ..$ Monetary..c.c..blood.: num [1:2, 1:2] 1187 1857 1197 1933
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ as.factor(train[, 5]): chr [1:2] "0" "1"
##   .. .. ..$ Monetary..c.c..blood.: NULL
##   ..$ Time..months.         : num [1:2, 1:2] 34.4 32.8 24.3 23.9
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ as.factor(train[, 5]): chr [1:2] "0" "1"
##   .. .. ..$ Time..months.         : NULL
##  $ levels : chr [1:2] "0" "1"
##  $ call   : language naiveBayes.default(x = train[, 1:4], y = as.factor(train[, 5]))
##  - attr(*, "class")= chr "naiveBayes"
```

```
bayespredict = cbind(predict(classifier,test[,-5]),test[,5])
```

```
a=ROC(bayespredict)
```



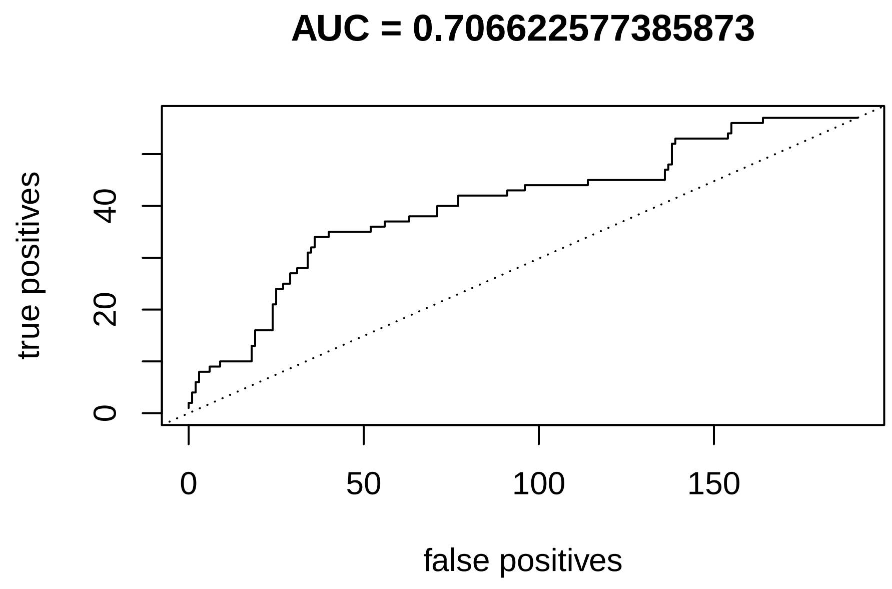


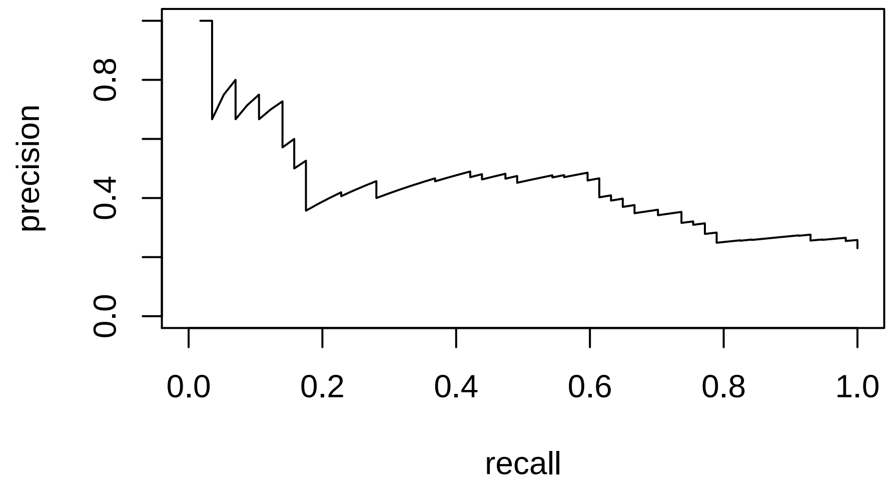
Well, it turns out that Bayesian statistics is not the answer to everything (Figure 4). About the same as the reduced logistic regression model. The curve is weirdly step-like, wonder what's going on there. Perhaps because NB is specifying categorical cutoffs in the continuous data?

"' More kNN. Figure 6: k=2, Figure 7: k=3, Figure 8: k=4, Figure 9: k=5.

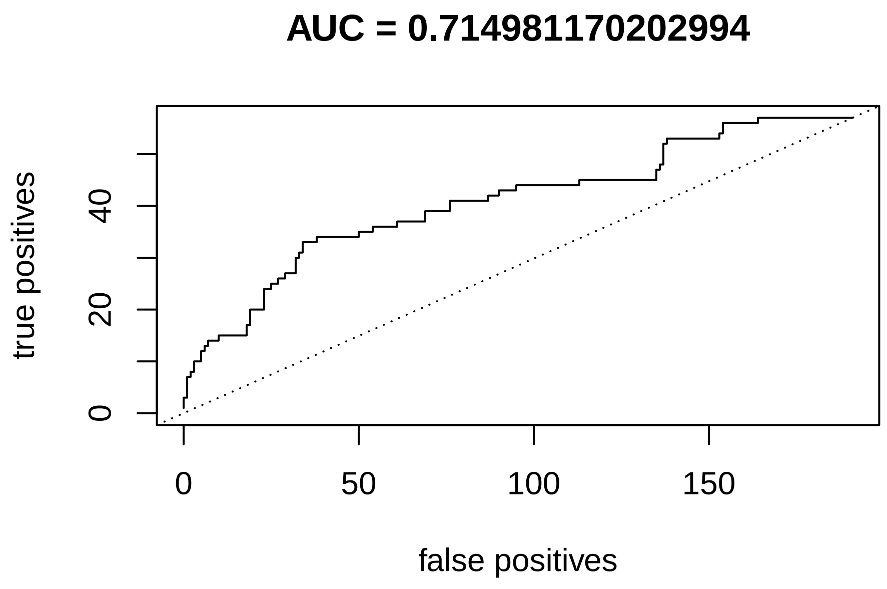
```
library(class)
nn2_pred = knn(train[,1:4],test=test[,1:4] ,cl=train[,5],k=2)
nn2_predict = cbind(nn2_pred,test[,5])
```

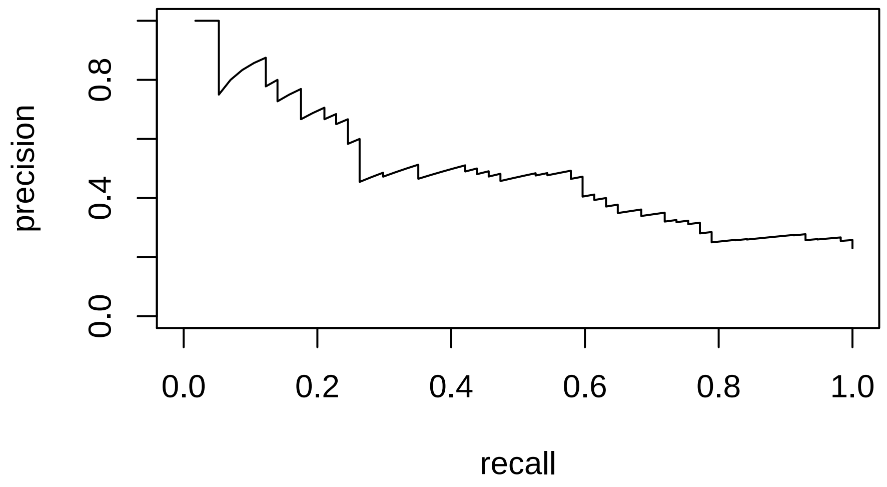
```
a=ROC(nn2_predict)
```



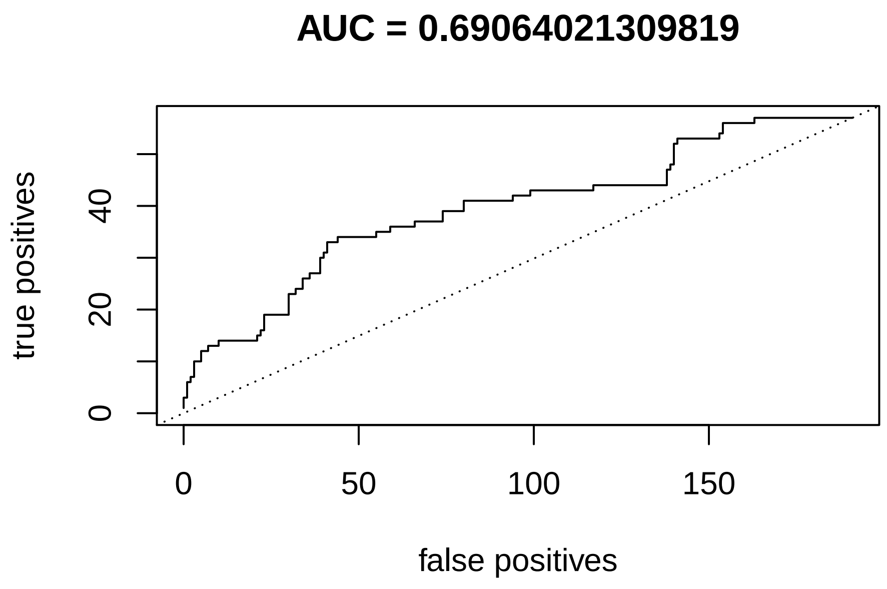


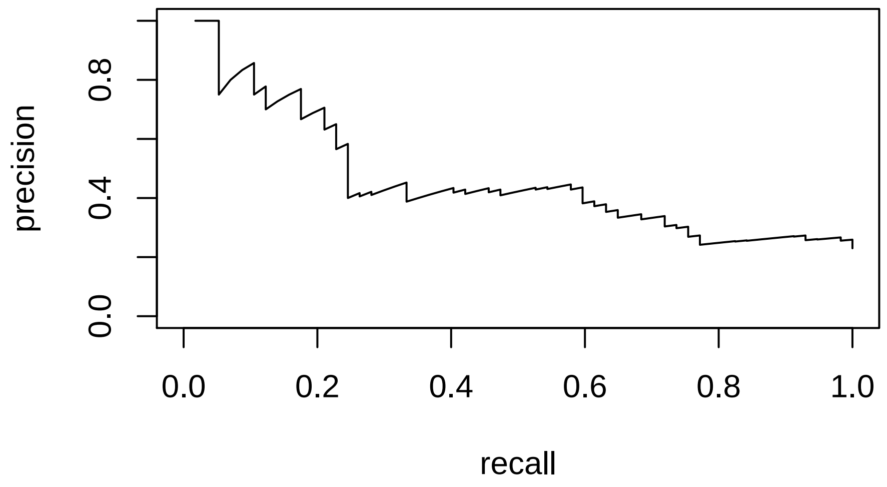
```
nn3_pred = knn(train[,1:4],test=test[,1:4] ,cl=train[,5],k=3)
nn3_predict = cbind(nn3_pred,test[,5])
a=ROC(nn3_predict)
```





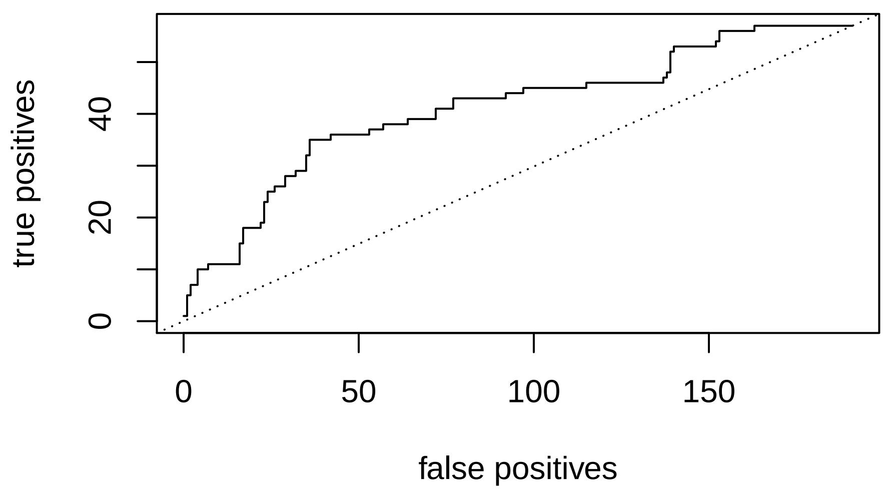
```
nn4_pred = knn(train[,1:4],cl=train[,5],test=test[,1:4] ,k=4)
nn4_predict = cbind(nn4_pred,test[,5])
a=ROC(nn4_predict)
```

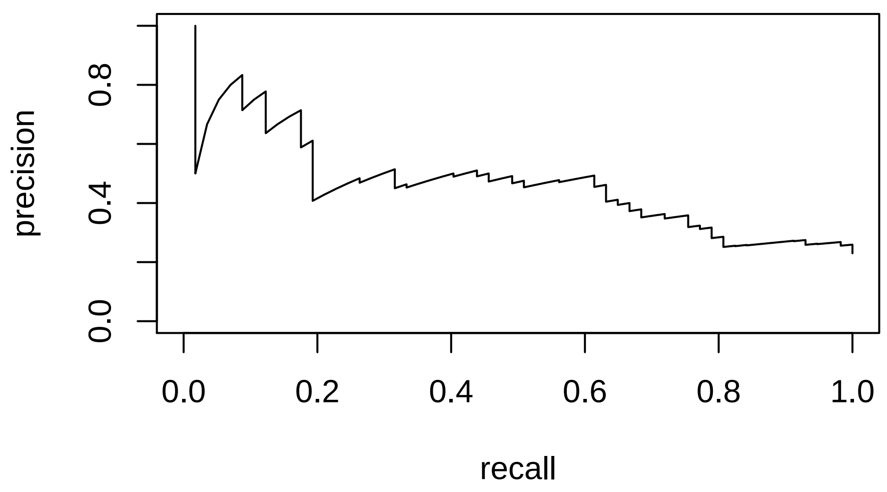




```
nn5_pred = knn(train[,1:4],test[,1:4],cl=train[,5] ,k=5)
nn5_predict = cbind(nn5_pred,test[,5])
a=ROC(nn5_predict)
```

AUC = 0.720492330302195


true positives
false positives


precision
recall