Study of breast cancer in Wisconsin

Neema Madayi Veetil

Thursday, May 7, 2015

This is an R Markdown document. The document explain the building of predictive model to determine whether the new study will be dealt appropriately.

The Data come from a study of breast cancer in Wisconsin. There are 681 cases of potentially cancerous tumors of which 238 are actually malignant. Determining whether a tumor is really malignant is traditionally determined by an invasive surgical procedure. The purpose of this study was to determine whether a new procedure called fine needle aspiration which draws only a small sample of tissue could be effective in determining tumor status.

A data frame with 681 observations on the following 10 variables.

Class - 0 if malignant, 1 if benign Adhes - marginal adhesion BNucl - bare nuclei Chrom - bland chromatin Epith - epithelial cell size Mitos - mitoses NNucl - normal nucleoli Thick - clump thickness UShap - cell shape uniformity USize - cell size uniformity

The predictor values are determined by a doctor observing the cells and rating them on a scale from 1 (normal) to 10 (most abnormal) with respect to the particular characteristic.

Source: Bennett, K.,P., and Mangasarian, O.L., Neural network training via linear programming. In P. M. Pardalos, editor, Advances in Optimization and Parallel Computing, pages 56-57. Elsevier Science, 1992

```
library(faraway)

## Warning: package 'faraway' was built under R version 3.1.3

data(wbca)
head(wbca)

##   Class Adhes BNucl Chrom Epith Mitos NNucl Thick UShap USize
## 1     1     1     1     3     2     1     1     5     1     1
## 2     1     5    10     3     7     1     2     5     4     4
## 3     1     1     2     3     2     1     1     3     1     1
## 4     1     1     4     3     3     1     7     6     8     8
## 5     1     3     1     3     2     1     1     4     1     1
## 6     0     8    10     9     7     1     7     8    10    10

View(wbca)
names(wbca)

##  [1] "Class" "Adhes" "BNucl" "Chrom" "Epith" "Mitos" "NNucl" "Thick"
##  [9] "UShap" "USize"
```

```r
## (a) Split the data set
v1 <- 1:nrow(wbca)
indx <- v1 %%3
threes <-v1[!indx]
others <- v1[!!indx]
#add a column of groups
wbca[which(rownames(wbca) %in% others), 'groups'] <- "True"
wbca[which(rownames(wbca) %in% threes), 'groups'] <- "False"
#factor the groups
g <- factor(wbca$groups)
#split the dataset by factor
splitted.data <- split(wbca, g)
#create train and test data
wbca.train <- splitted.data$"True"
wbca.train <- wbca.train[,-11]
wbca.test <- splitted.data$"False"
wbca.test <- wbca.test[,-11]
head(wbca.test)
```

```
##    Class Adhes BNucl Chrom Epith Mitos NNucl Thick UShap USize
## 3      1     1     2     3     2     1     1     3     1     1
## 6      0     8    10     9     7     1     7     8    10    10
## 9      1     1     1     1     2     5     1     2     1     1
## 12     1     1     1     2     2     1     1     2     1     1
## 15     0    10     9     5     7     4     5     8     5     7
## 18     1     1     1     3     2     1     1     4     1     1
```

```r
#fit a binomial regression with Class as response and the other nine
variables as predictors
fullmodel.fit <- glm(wbca.train$Class ~ .,family = binomial, data =
wbca.train)
summary(fullmodel.fit)
```

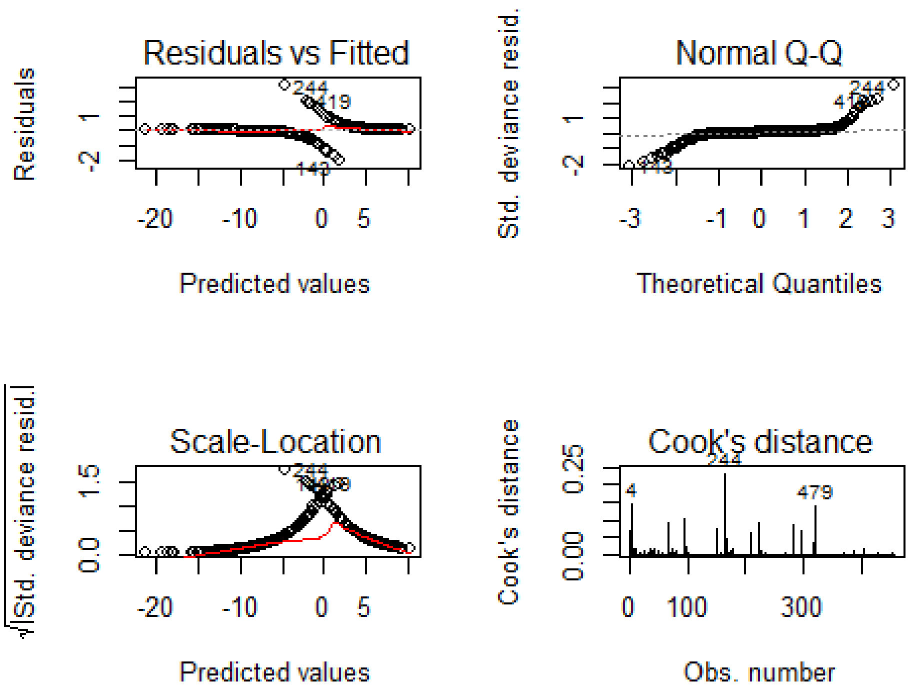
```
##
## Call:
## glm(formula = wbca.train$Class ~ ., family = binomial, data = wbca.train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.98138  -0.00954   0.03310   0.07084   3.07275
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  12.0244     2.0462   5.876 4.19e-09 ***
## Adhes        -0.4859     0.1555  -3.126  0.00177 **
## BNucl        -0.3732     0.1292  -2.888  0.00388 **
## Chrom        -0.6655     0.2536  -2.625  0.00868 **
## Epith         0.1779     0.2148   0.828  0.40744
## Mitos        -0.6075     0.5103  -1.190  0.23388
## NNucl        -0.5168     0.1828  -2.828  0.00469 **
## Thick        -0.6533     0.2044  -3.197  0.00139 **
## UShap        -0.5291     0.2612  -2.026  0.04280 *
## USize         0.2672     0.2320   1.152  0.24947
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 592.796  on 453  degrees of freedom
## Residual deviance:  57.651  on 444  degrees of freedom
## AIC: 77.651
##
## Number of Fisher Scoring iterations: 9
```

The following code is to check whether the fitted model is good fit. Test for Lack of Fit:

```
#########################################
#Test
#Comments:If your Null Deviance is really small, it means that the Null Model
explains the data pretty well.
#Likewise with your Residual Deviance.
#H0 = model fits well
#Ha = model does not fit well
pchisq(592.796, 453,lower=F)

## [1] 1.025295e-05

# H-L test of lack of fit
hosmerlem <- function (y, yhat, g = 10)
{
  cutyhat <- cut(yhat, breaks = quantile(yhat, probs = seq(0,
                                                           1, 1/g)),
include.lowest = T)
  obs <- xtabs(cbind(1 - y, y) ~ cutyhat)
  expect <- xtabs(cbind(1 - yhat, yhat) ~ cutyhat)
  chisq <- sum((obs - expect)^2/expect)
  P <- 1 - pchisq(chisq, g - 2)
  c("X^2" = chisq, Df = g - 2, "P(>Chi)" = P)
}
hosmerlem(y=wbca.train$Class, predict(fullmodel.fit, type="response"), g =
10)

##       X^2        Df   P(>Chi)
## 2.3534984 8.0000000 0.9682128
```

Residual Diagnostics Automated plots of residuals and fitted values

```
##################################################

##Automated plots:
par(mfrow=c(2,2))
for(i in 1:4) {
  plot(fullmodel.fit, which=i)
}
```



Let us now try stepwise variable selection to determine the best subsets of predictors. AIC criterion is decided. Code is as follows:

```
################################################################################
#######
# stepwise variable selection to determine the best subsets of predictors

step.aic <- step(fullmodel.fit, k=2, direction = 'both', trace=F)
step.aic$anova

##       Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1           NA        NA       444   57.65063 77.65063
## 2 - Epith   1 0.6895363       445   58.34016 76.34016
## 3 - USize   1 1.1956569       446   59.53582 75.53582

# best predictors are Adhes, BNucl, Chrom, Mitos, NNucl, Thick, UShap.
################################################################################
#######
```

The best predictor variables are:Adhes, BNucl, Chrom, Mitos, NNucl, Thick and UShap. After finding the best predictor, let us see if there is overdispersion or underdispersion. For testng dispersion, we can estimate the over dispersion parameter. From the code, we see that the estimate is less than 1, so it has underdispersion.

```
# fit the reduced model
reducedmodel.fit <- glm(wbca.train$Class ~
Adhes+BNucl+Chrom+Mitos+NNucl+Thick+UShap, family = binomial, data =
wbca.train)
summary(reducedmodel.fit)
```

```
##
## Call:
## glm(formula = wbca.train$Class ~ Adhes + BNucl + Chrom + Mitos +
##     NNucl + Thick + UShap, family = binomial, data = wbca.train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.03312  -0.01224   0.04042   0.08373   2.85056
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  11.5571     1.8285   6.321  2.6e-10 ***
## Adhes        -0.4249     0.1441  -2.949  0.00318 **
## BNucl        -0.3341     0.1187  -2.815  0.00487 **
## Chrom        -0.5963     0.2422  -2.462  0.01382 *
## Mitos        -0.5822     0.4872  -1.195  0.23207
## NNucl        -0.4192     0.1604  -2.614  0.00895 **
## Thick        -0.6037     0.1924  -3.138  0.00170 **
## UShap        -0.2943     0.2034  -1.447  0.14795
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 592.796  on 453  degrees of freedom
## Residual deviance:  59.536  on 446  degrees of freedom
## AIC: 75.536
##
## Number of Fisher Scoring iterations: 9
```

```
anova(reducedmodel.fit,fullmodel.fit,test = 'F', dispersion=0.3587)
```

```
## Analysis of Deviance Table
##
## Model 1: wbca.train$Class ~ Adhes + BNucl + Chrom + Mitos + NNucl + Thick
+
##     UShap
## Model 2: wbca.train$Class ~ Adhes + BNucl + Chrom + Epith + Mitos + NNucl
+
##     Thick + UShap + USize
##   Resid. Df Resid. Dev Df Deviance      F  Pr(>F)
## 1       446     59.536
## 2       444     57.651  2   1.8852 2.6278 0.07336 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##   Estimate the dispersion parameter
est.phi <- function(glmobj) {
  sum(
    residuals(glmobj, type="pearson")^2 / df.residual(glmobj)
  )
}
est.phi(fullmodel.fit)

## [1] 0.3587308

est.phi(reducedmodel.fit)

## [1] 0.2571319
```

Let us now fit both the models(full and reduced model) while accounting to underdispersion

```
# refit the full model while allowing for underdispersion
fullmodel.fit1 <- glm(wbca.train$Class ~ ., family=quasibinomial,
data=wbca.train)
summary(fullmodel.fit1)

##
## Call:
## glm(formula = wbca.train$Class ~ ., family = quasibinomial, data =
wbca.train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.98138  -0.00954   0.03310   0.07084   3.07275
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 12.02437    1.22555   9.811  < 2e-16 ***
## Adhes       -0.48593    0.09311  -5.219 2.77e-07 ***
## BNucl       -0.37319    0.07740  -4.821 1.96e-06 ***
## Chrom       -0.66549    0.15187  -4.382 1.47e-05 ***
## Epith        0.17791    0.12863   1.383 0.167325
## Mitos       -0.60752    0.30566  -1.988 0.047474 *
## NNucl       -0.51680    0.10947  -4.721 3.15e-06 ***
## Thick       -0.65330    0.12240  -5.337 1.51e-07 ***
## UShap       -0.52911    0.15645  -3.382 0.000783 ***
## USize        0.26723    0.13898   1.923 0.055148 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasibinomial family taken to be 0.3587316)
##
##     Null deviance: 592.796  on 453  degrees of freedom
## Residual deviance:  57.651  on 444  degrees of freedom
## AIC: NA
```

```
##
## Number of Fisher Scoring iterations: 9
```

```
# refit the subset model while allowing for underdispersion
reducedmodel.fit1 <- glm(wbca.train$Class ~
Adhes+BNucl+Chrom+Mitos+NNucl+Thick+UShap, family=quasibinomial,
data=wbca.train)
summary(reducedmodel.fit1)
```

```
##
## Call:
## glm(formula = wbca.train$Class ~ Adhes + BNucl + Chrom + Mitos +
##     NNucl + Thick + UShap, family = quasibinomial, data = wbca.train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.03312  -0.01224   0.04042   0.08373   2.85056
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.55714    0.92718  12.465  < 2e-16 ***
## Adhes       -0.42486    0.07305  -5.816 1.15e-08 ***
## BNucl       -0.33415    0.06019  -5.552 4.86e-08 ***
## Chrom       -0.59629    0.12282  -4.855 1.67e-06 ***
## Mitos       -0.58225    0.24706  -2.357  0.01887 *
## NNucl       -0.41921    0.08132  -5.155 3.82e-07 ***
## Thick       -0.60371    0.09755  -6.189 1.38e-09 ***
## UShap       -0.29430    0.10315  -2.853  0.00453 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasibinomial family taken to be 0.257132)
##
##     Null deviance: 592.796  on 453  degrees of freedom
## Residual deviance:  59.536  on 446  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 9
```

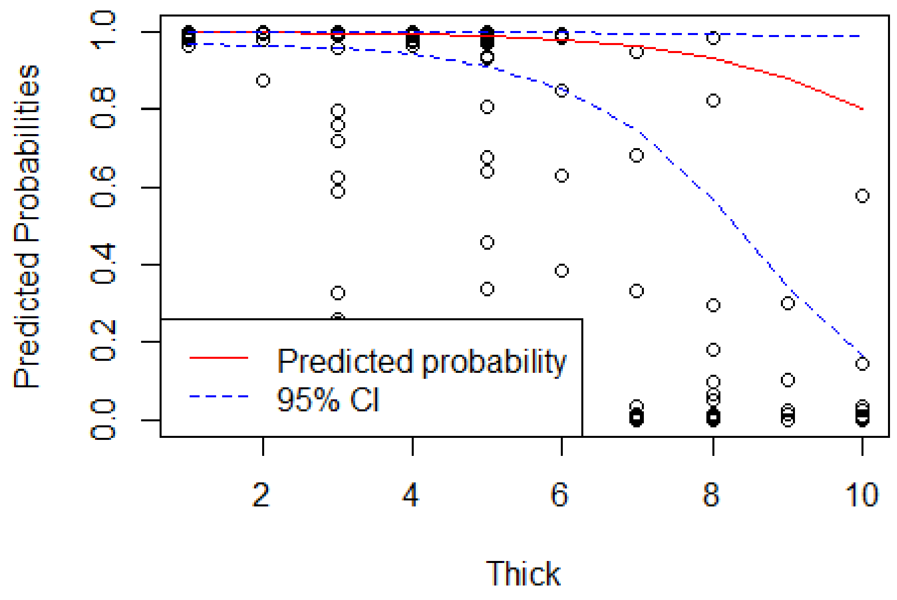
```
anova(reducedmodel.fit1,fullmodel.fit1, test = 'F')
```

```
## Analysis of Deviance Table
##
## Model 1: wbca.train$Class ~ Adhes + BNucl + Chrom + Mitos + NNucl + Thick
+
##     UShap
## Model 2: wbca.train$Class ~ Adhes + BNucl + Chrom + Epith + Mitos + NNucl
+
##     Thick + UShap + USize
##   Resid. Df Resid. Dev Df Deviance      F  Pr(>F)
## 1       446     59.536
## 2       444     57.651  2   1.8852 2.6276 0.07338 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

drop1(fullmodel.fit1, test="F")

## Single term deletions
##
## Model:
## wbca.train$Class ~ Adhes + BNucl + Chrom + Epith + Mitos + NNucl +
##     Thick + UShap + USize
##        Df Deviance  F value    Pr(>F)
## <none>      57.651
## Adhes   1   68.073  80.2695 < 2.2e-16 ***
## BNucl   1   67.373  74.8771 < 2.2e-16 ***
## Chrom   1   65.983  64.1750 1.011e-14 ***
## Epith   1   58.340   5.3105  0.021658 *
## Mitos   1   60.712  23.5783 1.665e-06 ***
## NNucl   1   67.538  76.1460 < 2.2e-16 ***
## Thick   1   71.162 104.0560 < 2.2e-16 ***
## UShap   1   61.450  29.2581 1.038e-07 ***
## USize   1   58.880   9.4716  0.002216 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

pchisq(0.95,25)

## [1] 3.425808e-14

### answer is reduced model because G^2 > F statistic. 2.6276>3.42E-14.
############################################################################
###########
```

Visualize the predictive ability by plot the response variable as a function of one predictor variable and overlay the others. Also with Bonferroni-corrected CI intervals for the mean.

```
par(mfrow=c(1,1))


plot(wbca.train$Thick, reducedmodel.fit$fitted.values, ylim=c(0,1),
     xlab="Thick",ylab="Predicted Probabilities")

newwbca <-
data.frame(Adhes=rep(median(wbca.train$Adhes),10),BNucl=rep(median(wbca.train
$BNucl),10),Chrom=rep(median(wbca.train$Chrom),10),

Mitos=rep(median(wbca.train$Mitos),10),NNucl=rep(median(wbca.train$NNucl),10)
,Thick=seq(from=1, to=10, length=10),
                      UShap=rep(median(wbca.train$UShap),10))
newwbca.predict <- predict(reducedmodel.fit, newdata=newwbca, se.fit=T,
type="response")
```

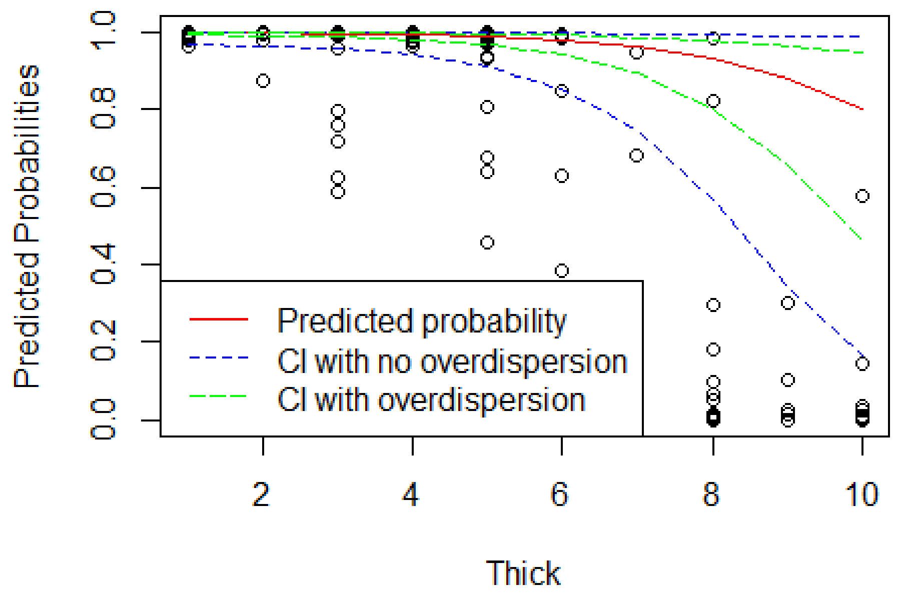
```
lines(newwbca$Thick, newwbca.predict$fit, col="red")

# -------------Bonferroni-corrected CI intervals for the mean-----------
newwbca.predictLink <- predict(reducedmodel.fit, newdata=newwbca, se.fit=T,
type="link")
L <- newwbca.predictLink$fit - qnorm(1-0.05/(2*10))*newwbca.predictLink$se
U <- newwbca.predictLink$fit + qnorm(1-0.05/(2*10))*newwbca.predictLink$se
lines(newwbca$Thick, 1/(1+exp(-L)), lty=2, col="blue")
lines(newwbca$Thick, 1/(1+exp(-U)), lty=2, col="blue")

legend("bottomleft", lty=c(1,2), col=c("red","blue"), c("Predicted
probability", "95% CI"))
```



The above plot is for the same for model with underdispersion, here we can see that the predictive line is below since it is underdispersion.

```
#with dispersion
plot(wbca.train$Thick, reducedmodel.fit$fitted.values, ylim=c(0,1),
     xlab="Thick",ylab="Predicted Probabilities")

newwbca <-
data.frame(Adhes=rep(median(wbca.train$Adhes),10),BNucl=rep(median(wbca.train
$BNucl),10),Chrom=rep(median(wbca.train$Chrom),10),

Mitos=rep(median(wbca.train$Mitos),10),NNucl=rep(median(wbca.train$NNucl),10)
,Thick=seq(from=1, to=10, length=10),
                      UShap=rep(median(wbca.train$UShap),10))
```

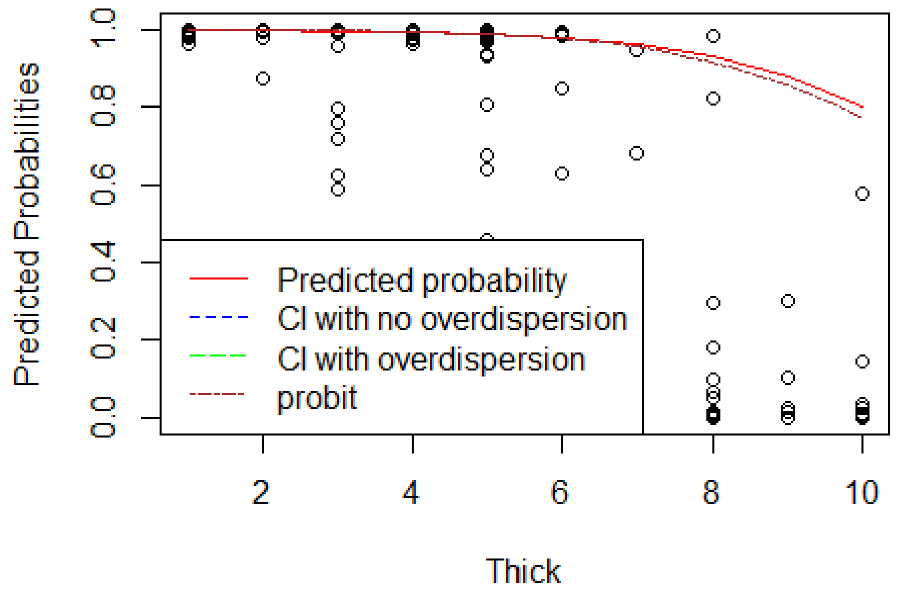
```r
newwbca.predict <- predict(reducedmodel.fit, newdata=newwbca, se.fit=T,
type="response")
lines(newwbca$Thick, newwbca.predict$fit, col="red")

# -------------Bonferroni-corrected CI intervals for the mean-----------
newwbca.predictLink <- predict(reducedmodel.fit, newdata=newwbca, se.fit=T,
type="link")
L <- newwbca.predictLink$fit - qnorm(1-0.05/(2*10))*newwbca.predictLink$se
U <- newwbca.predictLink$fit + qnorm(1-0.05/(2*10))*newwbca.predictLink$se
lines(newwbca$Thick, 1/(1+exp(-L)), lty=2, col="blue")
lines(newwbca$Thick, 1/(1+exp(-U)), lty=2, col="blue")

legend("bottomleft", lty=c(1,2), col=c("red","blue"), c("Predicted
probability", "95% CI"))

reducedmodel.fit1 <- glm(wbca.train$Class ~
Adhes+BNucl+Chrom+Mitos+NNucl+Thick+UShap, family=quasibinomial,
data=wbca.train)

newwbca.predictLink1 <- predict(reducedmodel.fit1, newdata=newwbca, se.fit=T,
type="link")
L1 <- newwbca.predictLink1$fit - qnorm(1-0.05/(2*10))*newwbca.predictLink1$se
U1 <- newwbca.predictLink1$fit + qnorm(1-0.05/(2*10))*newwbca.predictLink1$se
lines(newwbca$Thick, 1/(1+exp(-L1)), lty=5, col="green")
lines(newwbca$Thick, 1/(1+exp(-U1)), lty=5, col="green")
legend("bottomleft", lty=c(1,2,5), col=c("red","blue","green"), c("Predicted
probability","CI with no overdispersion", "CI with overdispersion"))
```



We will now visualize the predictive ability using an alternative link function:

```
################################################################################
###############


# -------------------------Alternative link function-------------------------
--

plot(wbca.train$Thick, reducedmodel.fit$fitted.values, ylim=c(0,1),
     xlab="Thick",ylab="Predicted Probabilities")

newwbca <-
data.frame(Adhes=rep(median(wbca.train$Adhes),10),BNucl=rep(median(wbca.train
$BNucl),10),Chrom=rep(median(wbca.train$Chrom),10),

Mitos=rep(median(wbca.train$Mitos),10),NNucl=rep(median(wbca.train$NNucl),10)
,Thick=seq(from=1, to=10, length=10),
                      UShap=rep(median(wbca.train$UShap),10))

newwbca.predict.logistic <- predict(reducedmodel.fit, newdata=newwbca,
se.fit=T, type="response")
lines(newwbca$Thick, newwbca.predict.logistic$fit, col="red")

# Compare with probit link
#fit the probit regression model
```

```
reducedmodel.fit2 <- glm(wbca.train$Class ~
Adhes+BNucl+Chrom+Mitos+NNucl+Thick+UShap, family=binomial(link="probit"),
data=wbca.train)

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

newwbca.predict.probit <- predict(reducedmodel.fit2, newdata=newwbca,
se.fit=T, type="response")
lines(newwbca$Thick, newwbca.predict.probit$fit, lty = 6,col="brown")


legend("bottomleft", lty=c(1,2,5,6), col=c("red", "blue","green","brown"),
       c("Predicted probability","CI with no overdispersion", "CI with
overdispersion", "probit"))
```



```
################################################################################
#############
```

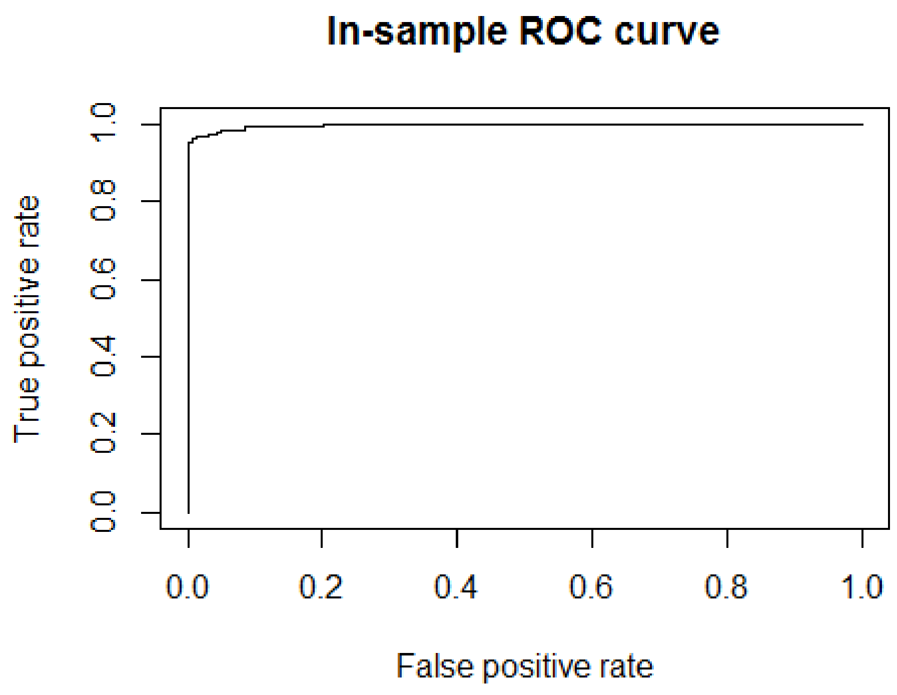
Let us now create a confusion matrix. The table is as folows:

```
predict.tumorstatus <- predict(reducedmodel.fit, newdata=wbca.train,
type="response")
table("Pred"=predict.tumorstatus > 0.5,"Class"=wbca.train$Class)

##        Class
## Pred      0   1
##   FALSE 156   7
##   TRUE    7 284
```

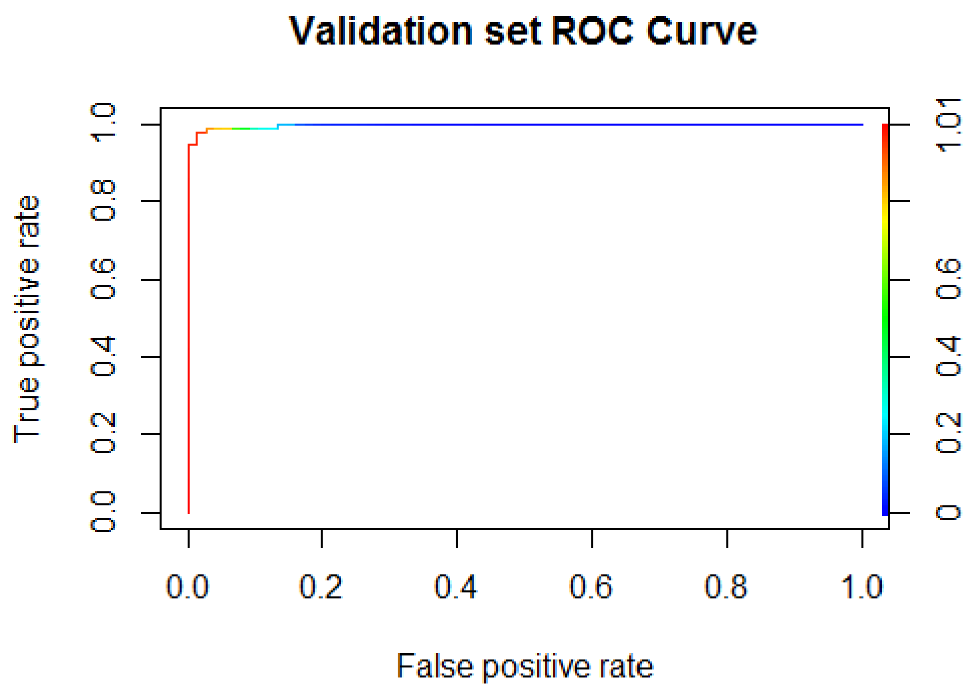
```
predict.tumorstatus <- predict(reducedmodel.fit, newdata=wbca.train,
type="response")
table("Pred"=predict.tumorstatus > 0.9,"Class"=wbca.train$Class)

##        Class
## Pred      0   1
##   FALSE 163  14
##   TRUE    0 277
```

Now using the package ROCR, we estimate the area under curve as well as plot the roc curve to check the predictive ability on training set and validation set.

```
library(ROCR)

## Warning: package 'ROCR' was built under R version 3.1.3

## Loading required package: gplots

## Warning: package 'gplots' was built under R version 3.1.3

## 
## Attaching package: 'gplots'
## 
## The following object is masked from 'package:stats':
## 
##     lowess

# calculate predicted probabilities on the same training set
scores <- predict(reducedmodel.fit, newdata=wbca.train, type="response")

# compare predicted probabilities to labels, for varying probability cutoffs
pred <- prediction(scores, labels=wbca.train$Class )
perf <- performance(pred, "tpr", "fpr")


# plot the ROC curve
plot(perf, type = "l",colorize=F, main="In-sample ROC curve")
```



```
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)

## [1] 0.9973225

# --------------Evaluate the predictive ability on the validation set--------
----
# make prediction on the validation dataset
scores <- predict(reducedmodel.fit, newdata=wbca.test, type="response")
pred <- prediction( scores, labels=wbca.test$Class )
perf <- performance(pred, "tpr", "fpr")

# overlay the line for the ROC curve
plot(perf, colorize=T, main ="Validation set ROC Curve")
```



```
# print out the area under the curve
unlist(attributes(performance(pred, "auc"))$y.values)

## [1] 0.9976316
```

The area under the curve for training set is 0.9976 and for validation set is 0.9976 which is close to 1.This shows that the predictive model has good ability of predicting for both training set and validation set. The area under the curve for the validation set is a good indicator that the model have good predictive ability.