GLM with binary response

Sami Cheong

1/22/2025

Heart disease example

What might affect the chance of getting heart disease? One of the earliest studies addressing this issue started in 1960 and used 3154 healthy men, aged from 39 to 59, from the San Francisco area. At the start of the study, all were free of heart disease. Eight and a half years later, the study recorded whether these men now suffered from heart disease along with many other variables that might be related to the chance of developing this disease.

We load a subset of this data (cigarette use, height, and heart disease) from the Western Collaborative Group Study described in Rosenman et al. (1975) and focus on just three of the variables in the dataset:

```
data(wcgs, package="faraway")
wcgs <- na.omit(wcgs)
summary(wcgs[,c("chd","height","cigs")])
```

```
##   chd           height           cigs
##  no :2885   Min.   :60.00   Min.   : 0.00
##  yes: 255   1st Qu.:68.00   1st Qu.: 0.00
##             Median :70.00   Median : 0.00
##             Mean   :69.78   Mean   :11.58
##             3rd Qu.:72.00   3rd Qu.:20.00
##             Max.   :78.00   Max.   :99.00
```

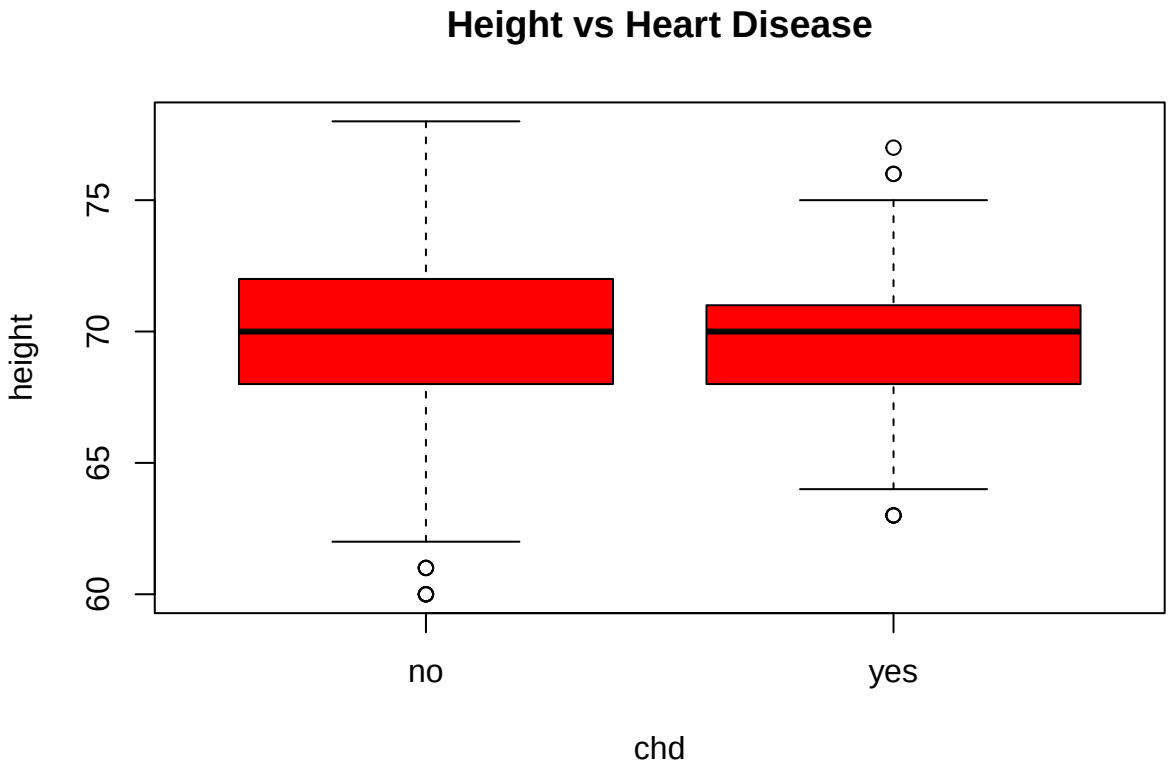
The variable `chd` indicates whether or not the person suffered from coronary heart disease, let's transform that to a binary variable

```
wcgs$y <- ifelse(wcgs$chd == 'yes',1,0)
```

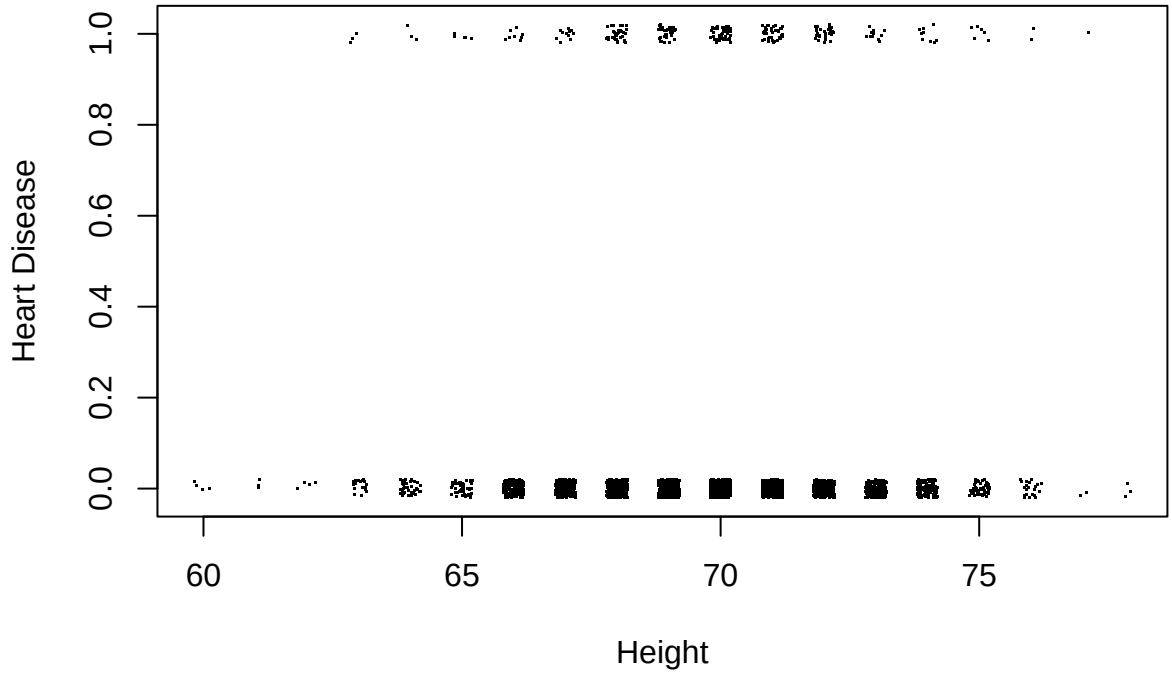
Exploratory data analysis:

Let's visualize the response and predictor variables in different dimensions to get a sense of how we should model it:

```
## boxplot view:
plot(height ~ chd, wcgs, col = 'red',
     main = 'Height vs Heart Disease' )
```



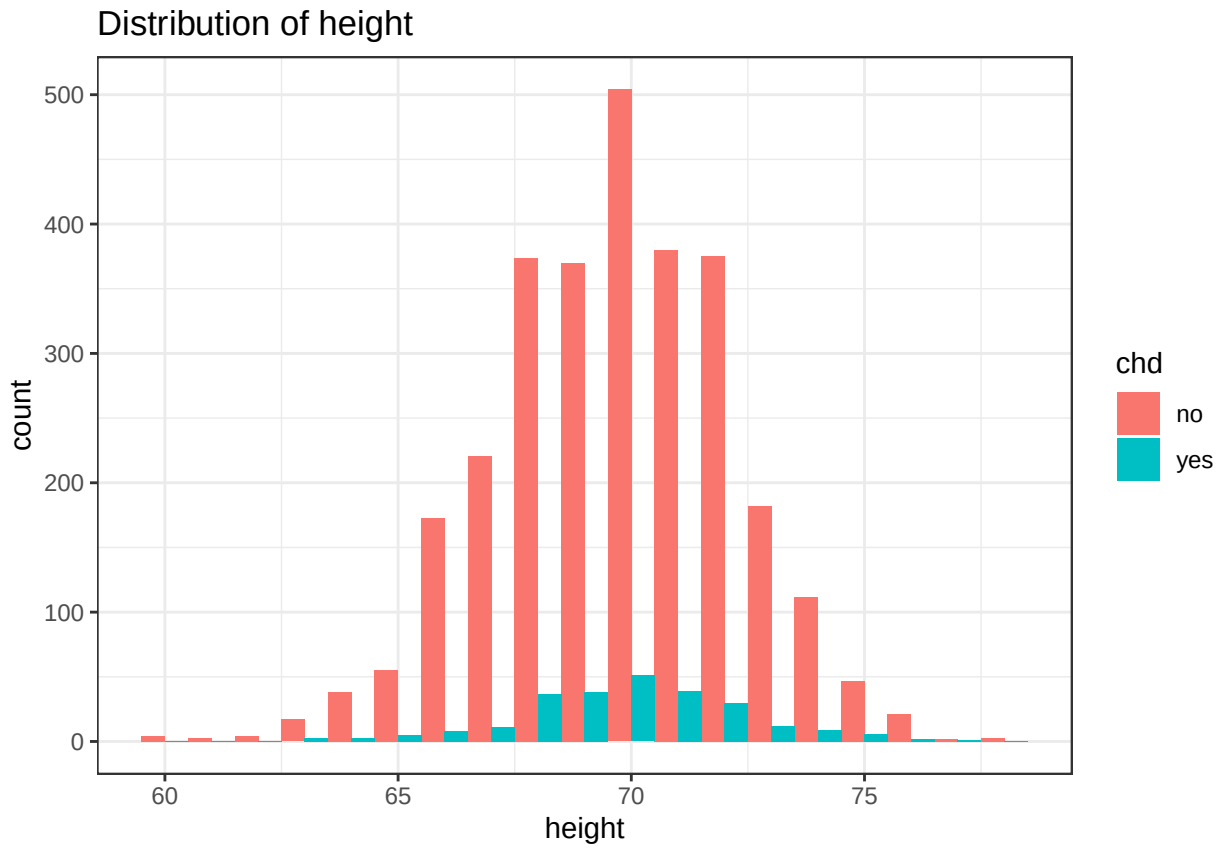

```
## jitter plot view
plot(jitter(y,0.1)~jitter(height),wcgs,
     xlab="Height", ylab="Heart Disease", pch=".")
```



We can also use `ggplot` to explore the visualization, for example here are some plots of height and cigarette consumption information of the subjects separated by heart disease status:

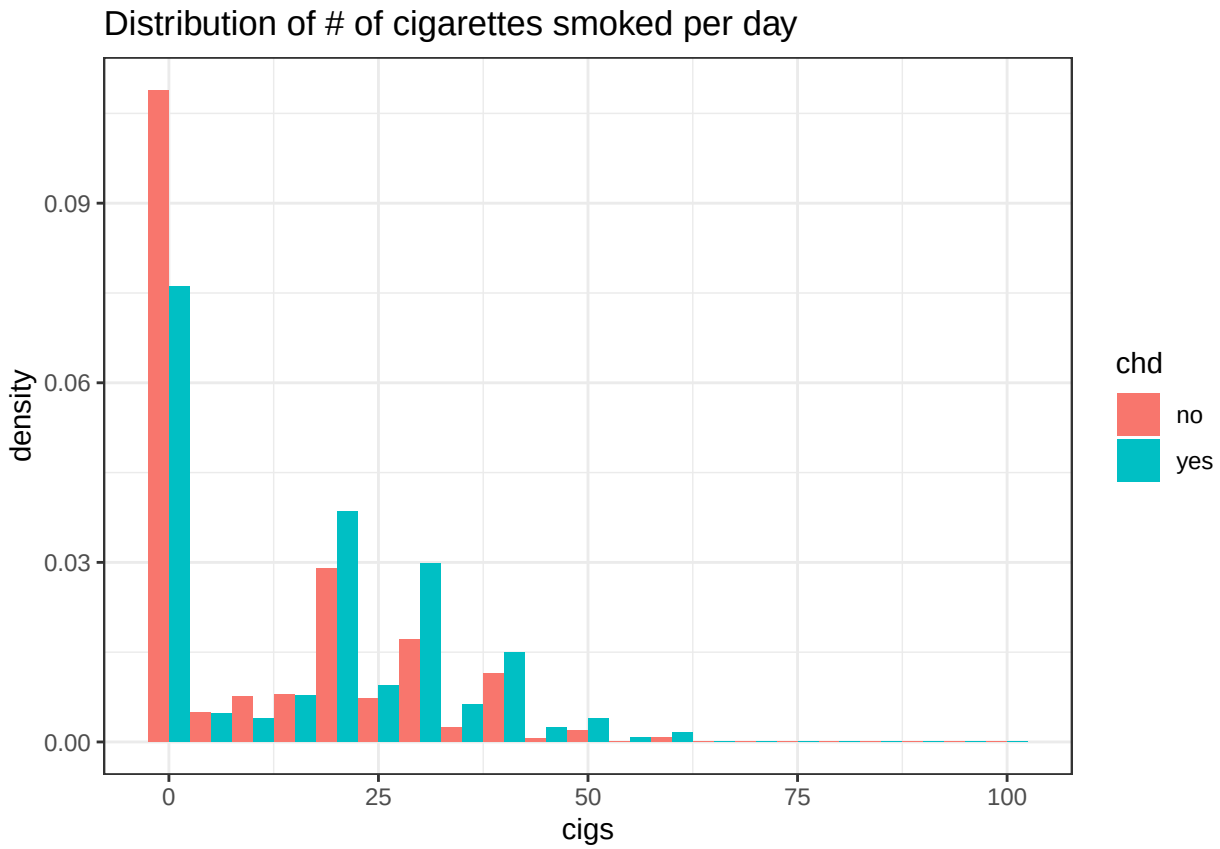
```
ggplot(wcgs, aes(x=height, fill=chd)) +
  geom_histogram(position="dodge", binwidth=1)+
```

```
  theme_bw()+ggtitle('Distribution of height')
```



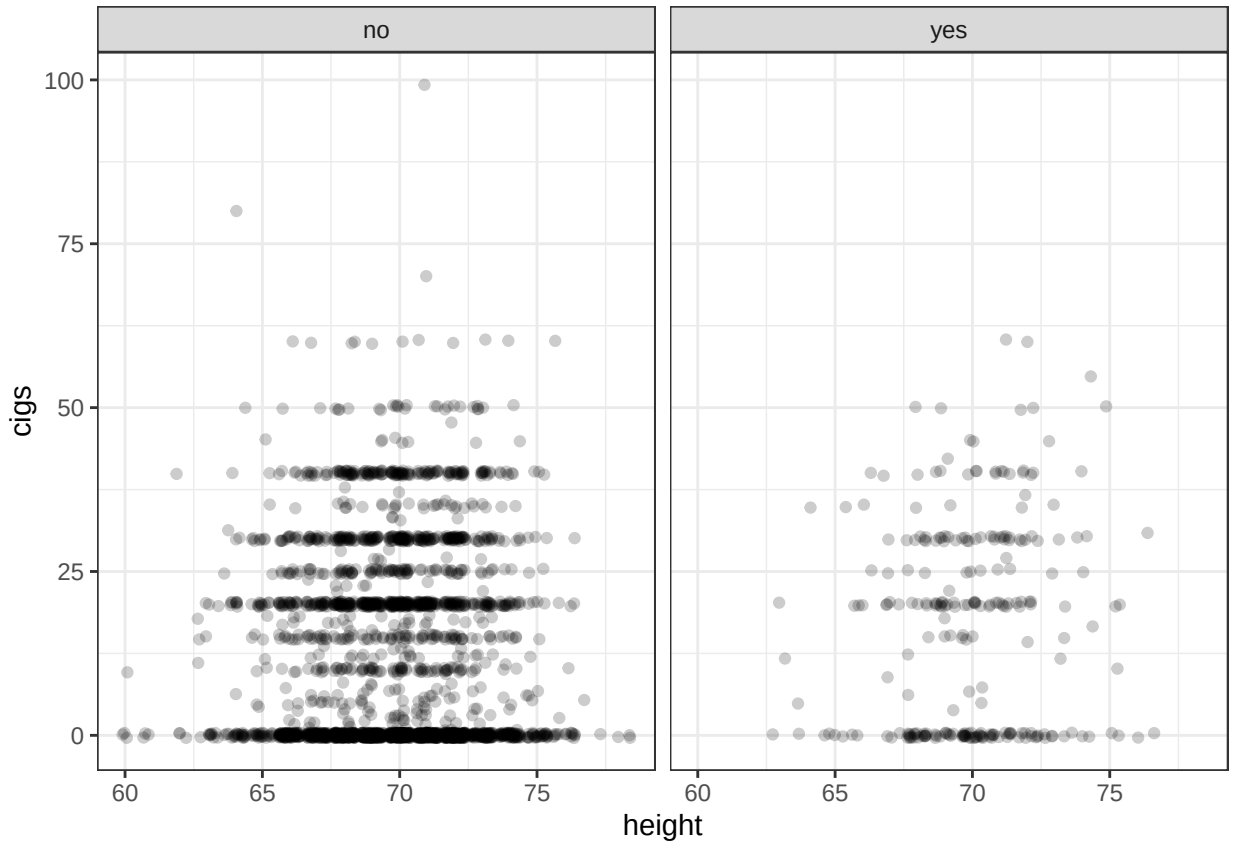

```
ggplot(wcgs, aes(x=cigs, fill=chd)) +
  geom_histogram(position="dodge", binwidth=5,
                 aes(y=..density..))+
  ggtitle('Distribution of # of cigarettes smoked per day')+theme_bw()
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



We can also separate the plots based on a chosen group using a `ggplot` feature called `facets`

```
ggplot(wcgs, aes(x=height,y=cigs))+
  geom_point(alpha=0.2, position=position_jitter())+
  facet_grid(~ chd)+theme_bw()
```



Analysis objective:

Our goal here is to predict the heart disease outcome for a given individual and also to explain the relationship between height, cigarette usage and heart disease.

Notice that for the same height and cigarette consumption, both outcomes can occur. It makes more sense then to model the response as a probability instead.

Model building :

Let y_i be a binary variable indicating presence or absence of heart disease (1, or 0). We would like to model it as the following :

$$p(\text{Heart disease for subject} i) = h(\beta_0 + \beta_1(\text{Height}) + \beta_2(\text{Cigarette use})$$

where $h = g^{-1}, g(\cdot)$ is the link function.

We can implement this in R as:

```
glmod <- glm(formula = chd ~ height + cigs,
             family = binomial(link = 'logit'),
             data = wcgs)
```

Understanding the model output:

Let's take a look at the summary of the model object `glmod`:

```
summary(glmod)
```

```
##
## Call:
## glm(formula = chd ~ height + cigs, family = binomial(link = "logit"),
##     data = wcgs)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.219165   1.852850  -2.277   0.0228 *
## height       0.021019   0.026495   0.793   0.4276
## cigs         0.023584   0.004059   5.810 6.24e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1769.2  on 3139  degrees of freedom
## Residual deviance: 1736.3  on 3137  degrees of freedom
## AIC: 1742.3
##
## Number of Fisher Scoring iterations: 5
```

Recall the model formula is the following:

$$p(\text{Heart disease for subject} i) = h(\beta_0 + \beta_1(\text{Height}) + \beta_2(\text{Cigarette use})$$

since we chose the logit function as the link function, this can either be expressed as

$$\log(\frac{p}{1-p}) = \beta_0 + \beta_1(\text{Height}) + \beta_2(\text{Cigarette use})$$

Model coefficients : in here the β values correspond to the change of the log odds of the response. Notice the range of the log odds is $(-\infty, \infty)$ vs $[0, 1]$

```
beta <- coef(glmod)
print(beta)
```

```
## (Intercept)      height        cigs
## -4.21916454  0.02101907  0.02358399
```

For example, keeping everything else fixed, increase 1 cigarette smoked per day is associated with increasing the log odds of having heart disease by 0.023, or 2.3%.

Another way to interpret the model result is to look at the odds instead:

$$\text{odds of heart disease} = e^{\beta_0} e^{\beta_1 \text{Height}} e^{\beta_2 \text{Cigs}}$$

The model output is the follows:

```
print(exp(beta))
```

```
## (Intercept)      height        cigs
##  0.01471093  1.02124153  1.02386430
```

This tells us that a unit increase in , for example, cigarettes smoked per day, increases the odds of heart disease by a factor of e^{β_1} or 1.023, which is again about an 2.3% increase.

Model inference

Null deviance: A low null deviance implies that the data can be modeled using only the intercept (without other predictors). If the null deviance is low, we should consider using fewer features for modeling the data.

Residual deviance: A low residual deviance implies that the model we have trained is appropriate.

There are several ways to compare models. We can use the deviance as a metric with the following example

```
D0<-1781.2
D1<-1749.0
dstar <- D0-D1
p_val<-1-pchisq(dstar,2)
print(p_val)
```

```
## [1] 1.01826e-07
```

We can also test the individual predictors by fitting models that drop these predictors and compute the difference in the deviance observed, here is an example testing the importance of the variance `height`

```
glmodc <- glm(chd ~ cigs, family = binomial, wcgs)
anova(glmodc,glmod, test="Chi")
```

```
## Analysis of Deviance Table
##
## Model 1: chd ~ cigs
## Model 2: chd ~ height + cigs
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      3138     1737.0
## 2      3137     1736.3  1  0.63111   0.4269
```

We can also test all predictors of the model in similar fashion using the `dropl` function

```
drop1(glmod,test="Chi")
```

```
## Single term deletions
##
## Model:
## chd ~ height + cigs
##        Df Deviance    AIC    LRT  Pr(>Chi)
## <none>      1736.3 1742.3
## height  1   1737.0 1741.0  0.631    0.4269
## cigs    1   1768.3 1772.3 32.011 1.533e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model testing by confidence interval

Wald-type : assumes the asymptotic distribution of $\hat{\beta}/se(\hat{\beta})$ follows the standard normal distribution

Profile likelihood: uses likelihood function to establish confidence interval

Example for our dataset here:

Recalling model output:

```
summary(glmod)$coefficients
```

```
##               Estimate  Std. Error    z value     Pr(>|z|)
## (Intercept) -4.21916454 1.852849603 -2.2771220 2.277894e-02
## height       0.02101907 0.026494843  0.7933268 4.275874e-01
```

```
## cigs         0.02358399 0.004059094  5.8101624 6.241228e-09
```

We can use the information above (Estimate, Std. Error) to construct a typical 95% confidence interval using `confint`. Take the coefficient for `height` as an example:

```
0.025 + c(-1,1) * 1.96 * 0.0263
```

```
## [1] -0.026548  0.076548
```

Alternatively, we can use the profile likelihood method:

```
confint(glmod)
```

```
##                   2.5 %      97.5 %
## (Intercept) -7.87233217 -0.60733146
## height      -0.03074108  0.07314496
## cigs         0.01557055  0.03150085
```

Model diagnostics

Two ways of computing model residuals:

```
## predictions on linear scale (Real line)
linpred <- predict(glmod)

## predictions on probability scale (0,1)
predprob <- predict(glmod, type="response")

print(head(linpred))
```

```
##         1         2         3         4         5         6
## -2.095173 -2.276150 -2.768849 -2.318188 -2.276150 -2.705792
```

```
print(head(predprob))
```

```
##          1          2          3          4          5          6
## 0.10956690 0.09311758 0.05903093 0.08962780 0.09311758 0.06263247
```

```
print(head(ilogit(linpred)))
```

```
##          1          2          3          4          5          6
## 0.10956690 0.09311758 0.05903093 0.08962780 0.09311758 0.06263247
```

Notice that the first one (linear predictions) are the values of $\eta_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i}$, while the second one is $h(\eta_i), h = g^{-1}$. In this case, $g = logit()$ so h is the inverse of the logit function (in the Faraway package that is `ilogit`)

Let's take a look at the raw residuals $y_i - \hat{p}_i$

```
rawres <- wcgs$y - predprob

print(head(rawres))
```

```
##           1           2           3           4           5           6
## -0.10956690 -0.09311758 -0.05903093 -0.08962780  0.90688242 -0.06263247
```

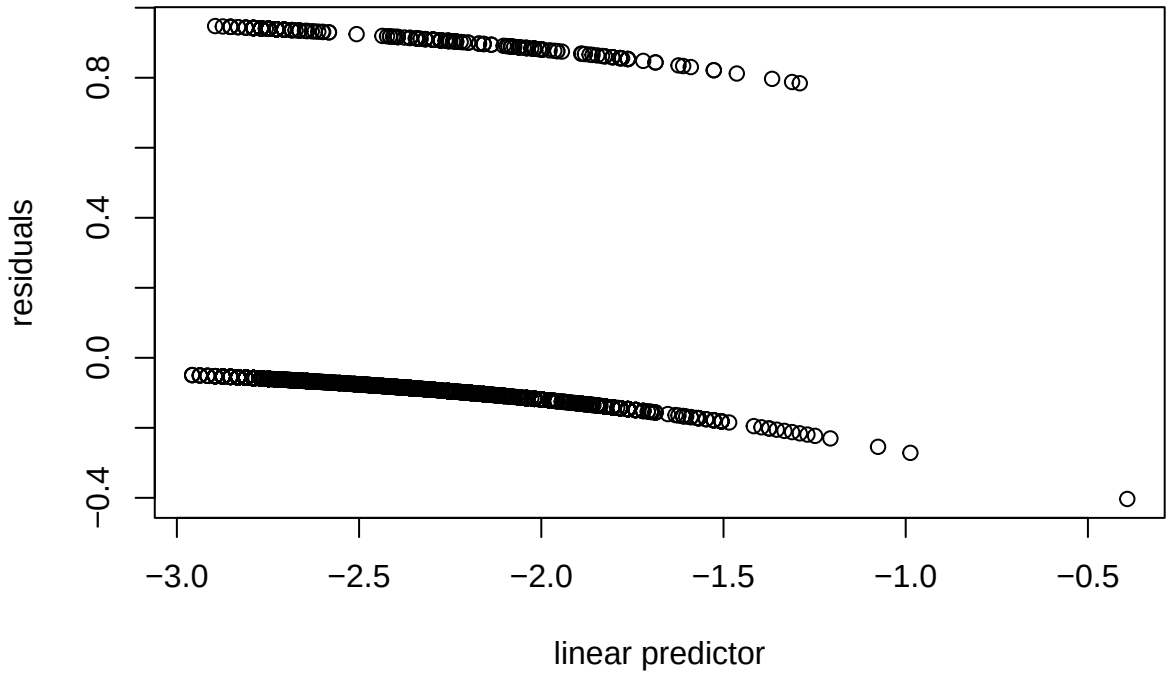
Alternatively, can also get them using the `residuals` function

```
print(head(residuals(glmod, type="response")))
```

```
##           1           2           3           4           5           6
## -0.10956690 -0.09311758 -0.05903093 -0.08962780  0.90688242 -0.06263247
```

Let's take a first look at the residual plots:

```
plot(rawres ~ linpred, xlab="linear predictor", ylab="residuals")
```



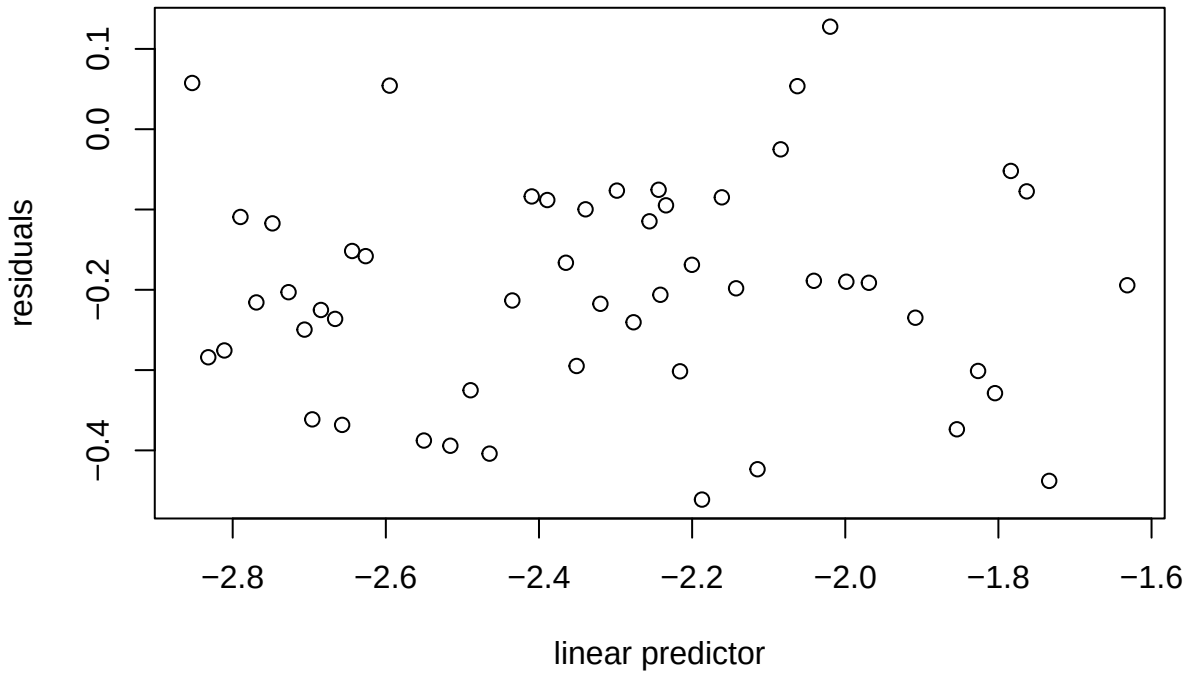
We now construct a more useful residuals plot by grouping the residuals into bins where the bins are based on similar predictor values.

The choice of the number of bins depends on the size of the dataset.

We choose 100 bins so that we have roughly 30 observations per bin. Some effort is required to construct this plot.

The `dplyr` package of Wickham and Francois (2015) is useful for this task. First, we add the residuals and linear predictor into the data frame.

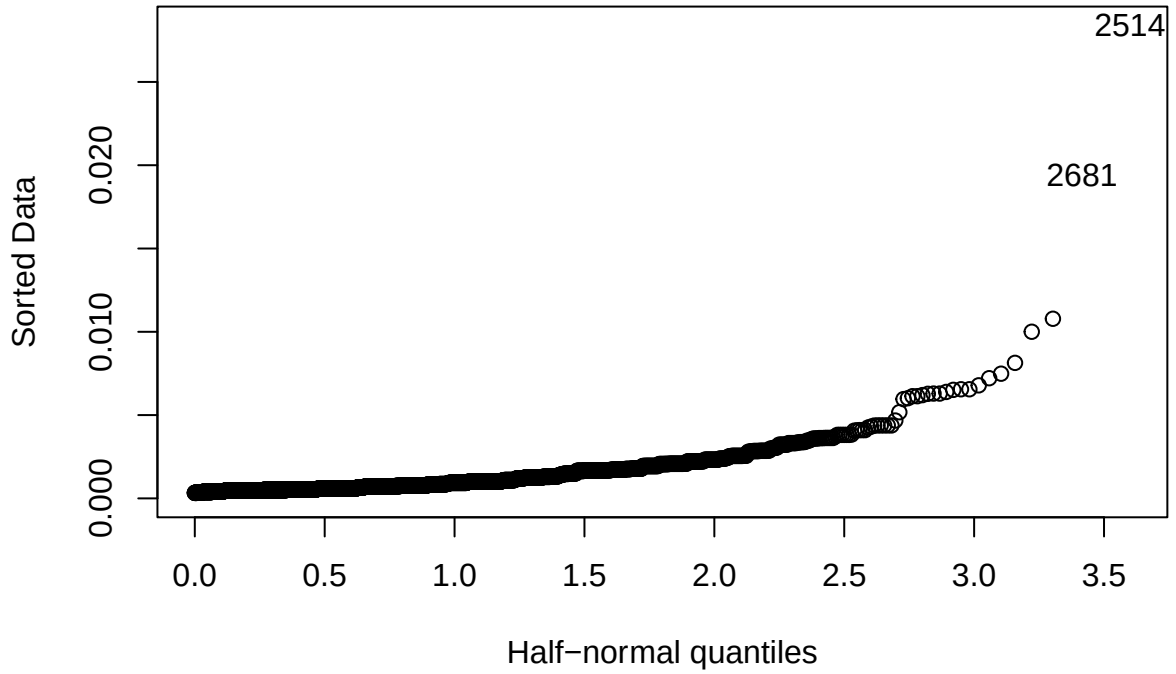
```
library(dplyr)
wcgs <- mutate(wcgs, residuals=residuals(glmod), linpred=predict(glmod))

## create bins for the values:

gdf <- group_by(wcgs, cut(linpred, breaks=unique(quantile(linpred,(1:100)/101))))

## summarize by mean of the residuals and predictors
diagdf <- summarise(gdf, residuals=mean(residuals), linpred=mean(linpred))

plot(residuals ~ linpred, diagdf, xlab="linear predictor")
```



Detecting unusual cases via hat values:

Here's an example using half-normal plot

```
halfnorm(hatvalues(glmod))
```



Model selection

There are a few different methods for model selection. One common approach is the backward algorithm, which can be implemented using the `drop1` function

```r
## create a new variable called bmi:
wcgs$bmi <- with(wcgs, 703*wcgs$weight/(wcgs$height^2))
glmod1 <- glm(chd ~ age + height + weight +bmi
              + sdp + dbp + chol +dibep +
                cigs +arcus, family=binomial, wcgs)


drop1(glmod1,test = 'Chisq')
```

```
## Single term deletions
##
## Model:
## chd ~ age + height + weight + bmi + sdp + dbp + chol + dibep +
##     cigs + arcus
##        Df Deviance    AIC    LRT  Pr(>Chi)
## <none>      1569.2 1591.2
## age     1   1593.8 1613.8 24.618 6.989e-07 ***
## height  1   1569.5 1589.5  0.285  0.593689
## weight  1   1569.3 1589.3  0.099  0.753181
## bmi     1   1569.5 1589.5  0.258  0.611578
## sdp     1   1577.0 1597.0  7.826  0.005151 **
## dbp     1   1569.2 1589.2  0.011  0.916620
## chol    1   1620.0 1640.0 50.735 1.057e-12 ***
## dibep   1   1590.5 1610.5 21.333 3.860e-06 ***
## cigs    1   1592.2 1612.2 23.013 1.609e-06 ***
## arcus   1   1571.3 1591.3  2.098  0.147446
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
## drop height, bmi, weight, dbp, arcus
glmod2 <- glm(chd ~ age  + sdp + chol +dibep + cigs ,
              family=binomial, wcgs)
drop1(glmod2,test = 'Chisq')
```

```
## Single term deletions
##
## Model:
## chd ~ age + sdp + chol + dibep + cigs
##        Df Deviance    AIC    LRT  Pr(>Chi)
## <none>      1579.5 1591.5
## age     1   1605.3 1615.3 25.885 3.625e-07 ***
## sdp     1   1603.4 1613.4 23.917 1.006e-06 ***
## chol    1   1632.7 1642.7 53.276 2.898e-13 ***
## dibep   1   1602.2 1612.2 22.728 1.866e-06 ***
## cigs    1   1602.0 1612.0 22.567 2.030e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

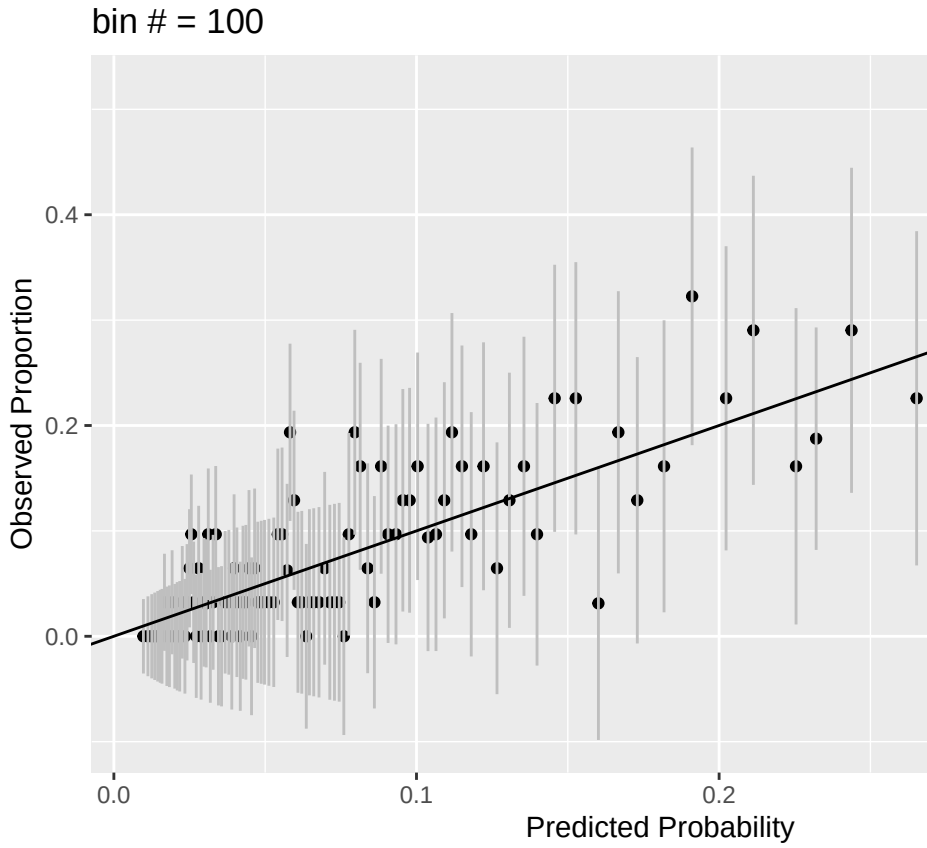
```r
### try subset selection using AIC:

glmod3<-step(glmod1,trace = 0)
summary(glmod3)
```

```
##
## Call:
```

```
## glm(formula = chd ~ age + height + bmi + sdp + chol + dibep +
##     cigs + arcus, family = binomial, data = wcgs)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -15.299983   2.294134  -6.669 2.57e-11 ***
## age            0.061590   0.012397   4.968 6.76e-07 ***
## height         0.050161   0.027824   1.803   0.0714 .
## bmi            0.060385   0.026599   2.270   0.0232 *
## sdp            0.017728   0.004155   4.267 1.98e-05 ***
## chol           0.010709   0.001529   7.006 2.45e-12 ***
## dibepB        -0.657616   0.145898  -4.507 6.56e-06 ***
## cigs           0.021041   0.004262   4.936 7.96e-07 ***
## arcuspresent   0.210998   0.143718   1.468   0.1421
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1769.2  on 3139  degrees of freedom
## Residual deviance: 1569.3  on 3131  degrees of freedom
## AIC: 1587.3
##
## Number of Fisher Scoring iterations: 6
```

Checking model fit

```
## create a new column in the dataset called predprob
## which is the output of the logistic regression model
wcgs <- mutate(wcgs, linpred = fitted(glmod3), predprob=predict(glmod3,type="response"))
## create bins and group the model ouptut by them
gdf <- group_by(wcgs, cut(linpred, breaks=unique(quantile(linpred,(1:100)/101))))
hldf <- summarise(gdf, y=sum(y),
                  ppred=mean(predprob),
                  count=n())%>%
  mutate(se.fit=sqrt(ppred*(1-ppred)/count))

ggplot(hldf,aes(x=ppred,
                y=y/count,
                ymin=y/count-2*se.fit,
                ymax=y/count+2*se.fit))+geom_point()+
  geom_linerange(color=grey(0.75))+
  geom_abline(intercept=0,slope=1)+xlab("Predicted Probability")+ylab("Observed Proportion")+ggtitle('b
```



Hosmer-Lemeshow method (Binning)

```
### H-L statistic and degrees of freedom:

hlstat <- with(hldf,
               sum( (y-count*ppred)^2/(count*ppred*(1-ppred))))
c(hlstat, nrow(hldf))
```

```
## [1]  99.11423 100.00000
```

```
## p-value of the observed H-L stat:

1-pchisq(63.212, 56-1)
```

```
## [1] 0.2089823
```

```
wcgs <- mutate(wcgs,
               predout = ifelse(predprob < 0.5, "no", "yes"))
xtabs( ~ chd + predout, wcgs)
```

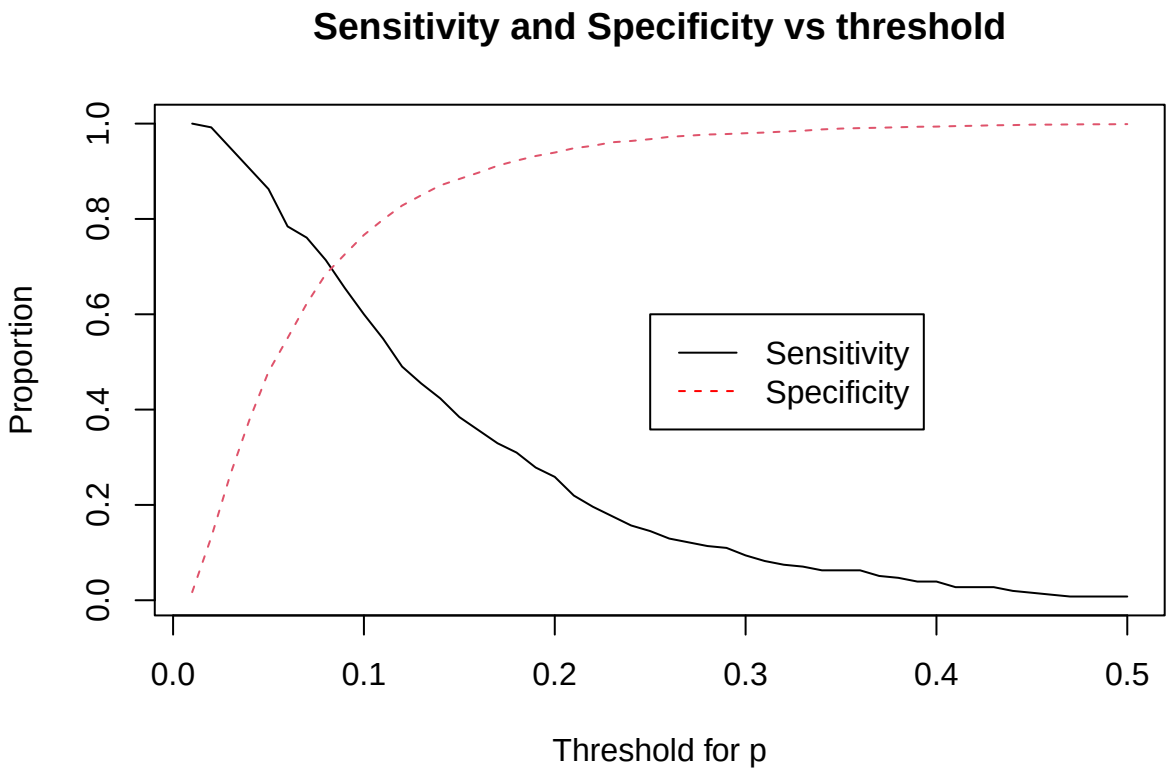
Contingency table

```
##      predout
## chd     no  yes
##   no  2882    3
##   yes  253    2
```

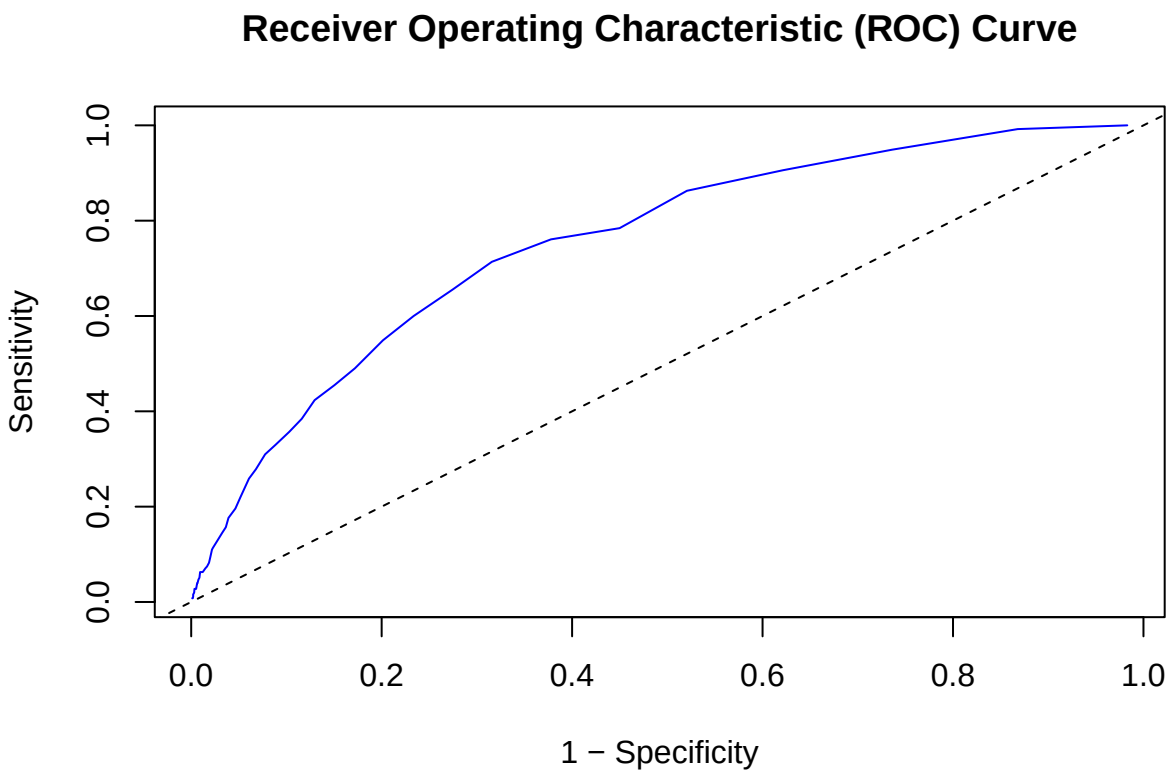
ROC curve Testing an array of threshold for the predicted probability of heart disease, then see how it changes the true negative (specificity) vs true positive (sensitivity)

```
thresh <- seq(0.01,0.5,0.01)
Sensitivity <- numeric(length(thresh))
Specificity <- numeric(length(thresh))
for(j in seq(along=thresh)){
  pp <- ifelse(wcgs$predprob < thresh[j],
               "no","yes")
  xx <- xtabs( ~ chd + pp, wcgs)
  Specificity[j] <- xx[1,1]/(xx[1,1]+xx[1,2])
       Sensitivity[j] <- xx[2,2]/(xx[2,1]+xx[2,2])
}
```

```
matplot(thresh,cbind(Sensitivity,Specificity),
        type="l",xlab="Threshold for p",
        ylab="Proportion",lty=1:2,
        main ='Sensitivity and Specificity vs threshold')
legend(0.25,0.6,lty=1:2,
       col = c('black','red'),
       legend = c('Sensitivity','Specificity'))
```




```
plot(1-Specificity,Sensitivity,type="l",
     col='blue',
     main = 'Receiver Operating Characteristic (ROC) Curve' )
abline(0,1,lty=2)
```


Receiver Operating Characteristic (ROC) Curve
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
1 − Specificity