Lee, Daniel
18379784
Stat 230A
4/28/17

Problem Set 6

Problem 1.

$$f(y|\theta)=\exp[a(y)b(\theta)+c(\theta)+d(y)]$$

Note the following:

① $\int f(y|\theta)dy=1$

② $\frac{d}{d\theta}\int f(y|\theta)dy=\frac{d}{d\theta}1=0$

③ $\int \frac{df(y|\theta)}{d\theta}dy=0$

④ $\int \frac{d^2f(y|\theta)}{d\theta^2}dy=0$

Using these, calculate $E[a(Y)]$:

$$f(y|\theta)=\exp[a(y)b(\theta)+c(\theta)+d(y)]$$

$$\frac{df(y|\theta)}{d\theta}=[a(y)b'(\theta)+c'(\theta)]f(y|\theta)$$

$\int [a(y)b'(\theta)+c'(\theta)]f(y|\theta)dy=0$ (using ③)

$$= b'(\theta)\int a(y)f(y|\theta)dy + \int c'(\theta)f(y|\theta)dy$$

$$= b'(\theta)E(a(y)) + c'(\theta) = 0 \quad (\text{using } ①)$$

$$\Rightarrow \boxed{E[a(Y)] = -\frac{c'(\theta)}{b'(\theta)}}$$

Var $[a(Y)]$ is derived as follows:

$$\frac{d^2 f(y|\theta)}{d\theta^2} = [a(y)b''(\theta) + c''(\theta)]f(y|\theta) + [a(y)b'(\theta) + c'(\theta)]^2 f(y|\theta)$$

$$[a(y)b'(\theta) + c'(\theta)]^2 f(y|\theta) = [b'(\theta)]^2\{a(y) - E(a(Y))\}^2 f(y|\theta)$$

$$\left(\text{using } E[a(Y)] = -\frac{c'(\theta)}{b'(\theta)}\right)$$

Therefore,

$$\int \frac{d^2 f(y|\theta)}{d\theta^2} dy = b''(\theta)E[a(Y)] + c''(\theta) + [b'(\theta)]^2 \operatorname{var}[a(Y)] = 0$$

since $\int \{a(y) - E[a(Y)]\}^2 f(y|\theta)dy = \operatorname{Var}[a(Y)]$ by definition.

Solving for var $[a(Y)]$:

$$\text{Var}[a(Y)] = \frac{-b''(\theta)E[a(Y)] - c''(\theta)}{[b'(\theta)]^2}$$

Substitute $E[a(Y)] = -\dfrac{c'(\theta)}{b'(\theta)}$:

$$\text{Var}[a(Y)] = \frac{+b''(\theta)\dfrac{c'(\theta)}{b'(\theta)} - c''(\theta)}{(b'(\theta))^2} * \frac{b'(\theta)}{b'(\theta)}$$

$$\boxed{\text{Var}[a(Y)] = \frac{b''(\theta)c'(\theta) - c''(\theta)b'(\theta)}{[b'(\theta)]^3}}$$

2. Specify the three elements when using GLM in the following scenarios:

a. Linear model:

$$Y \sim N(\mu, \sigma^2)$$

$$f_Y(y|\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(y-\mu)^2\right)$$

Link function g is the identity;

Linear predictor $\eta = X\beta$ is

$\eta = g(\mu) = \mu = X\beta$. This is the mean function.

2b. Poisson model:

$Y \sim \text{Poisson}(\mu)$

$$f_Y(y|\theta) = \frac{e^{-\mu}\mu^y}{y!}, \quad y = 0, 1, 2, \ldots$$

Link function g is $\log$

Linear predictor $\eta = X\beta$ is

$\eta = g(\mu) = \log(\mu) = X\beta$. This is the mean function.

2c. Categorical Logistic Regression

$Y \sim \text{Categorical}(\pi_1, \ldots, \pi_c)$,

where c is the number of categories.

$$f_Y(y|\theta) = \pi_1^{y_1} \pi_2^{y_2} \cdots \pi_c^{y_c}$$

such that $\pi_1 + \pi_2 + \cdots + \pi_c = 1$

and $y_i := [y = i]$.

Link function g is the logit function.

Linear predictor $\eta = X\beta$ is

$$\eta = \ln\left(\frac{\mu}{1-\mu}\right) = X\beta$$

where the mean function μ is

$$\mu = \frac{\exp(X\beta)}{1+\exp(X\beta)} = \frac{1}{1+\exp(-X\beta)}$$

2d. Multinomial Logistic Regression:

$Y \sim \text{Multinom}(n, \pi_1, \dots, \pi_c)$, where c is the number of categories

$$f_Y(y|\theta) = \frac{n!}{y_1! \, y_2! \cdots y_c!} \pi_1^{y_1} \pi_2^{y_2} \cdots \pi_c^{y_c}$$

such that $\pi_1 + \pi_2 + \cdots + \pi_c = 1$

and $y = (y_1, \dots, y_c)$.

Link function g is the generalized logit function.

Linear predictor $\eta = X\beta$ is the following:

Let $y_i = (y_{i1}, \dots, y_{i,c-1})^T$.

Then $y_{ic} = 1 - (y_{i1} + \cdots + y_{i,c-1})$.

Let $\mu_i = (\pi_{i1}, \dots, \pi_{i,c-1})^T$.

$\eta = X\beta$ is $g(\mu_i) = X_i\beta$, $i = 1, \dots, n$, where g is a vector of link functions. That is,

$$g_j(\mu_i) = \log \frac{\mu_{ij}}{1 - (\mu_{i1} + \cdots + \mu_{i,c-1})}, \quad j = 1, \dots, p$$

(p is number of predictors $x_1, \dots, x_p$)

2d. continued.

In other words, the mean function is

$$\mu = \frac{\exp(X\beta)}{1+\exp(X\beta)} = \frac{1}{1+\exp(-X\beta)}$$

Homework 6

Daniel Lee

April 28, 2017

Problem 2

2.1. In the *binary* dataset, there are two covariates and the response in binary. Do the following steps:

- Linear regression
- Logistic regression
- Probit regression

```
setwd("C:/Users/Daniel/Desktop/Spring 2017/Stat 230A/hw6")

load("binary.RData")
class(binary)
```

```
## [1] "data.frame"
```

```
str(binary)
```

```
## 'data.frame':    20 obs. of  3 variables:
##  $ x1: num  0.0979 -0.5493 0.185 0.5832 0.0961 ...
##  $ x2: num  -0.5002 0.00835 -0.79325 2.06502 0.42724 ...
##  $ y : logi  FALSE TRUE TRUE TRUE TRUE FALSE ...
```

```
dim(binary)
```

```
## [1] 20  3
```

Here, the outcome variable is binary, so either the logistic regression or the probit regression would make more sense since linear regression can produce fitted values of the probability of $Y = 1$ that are not between zero and one.

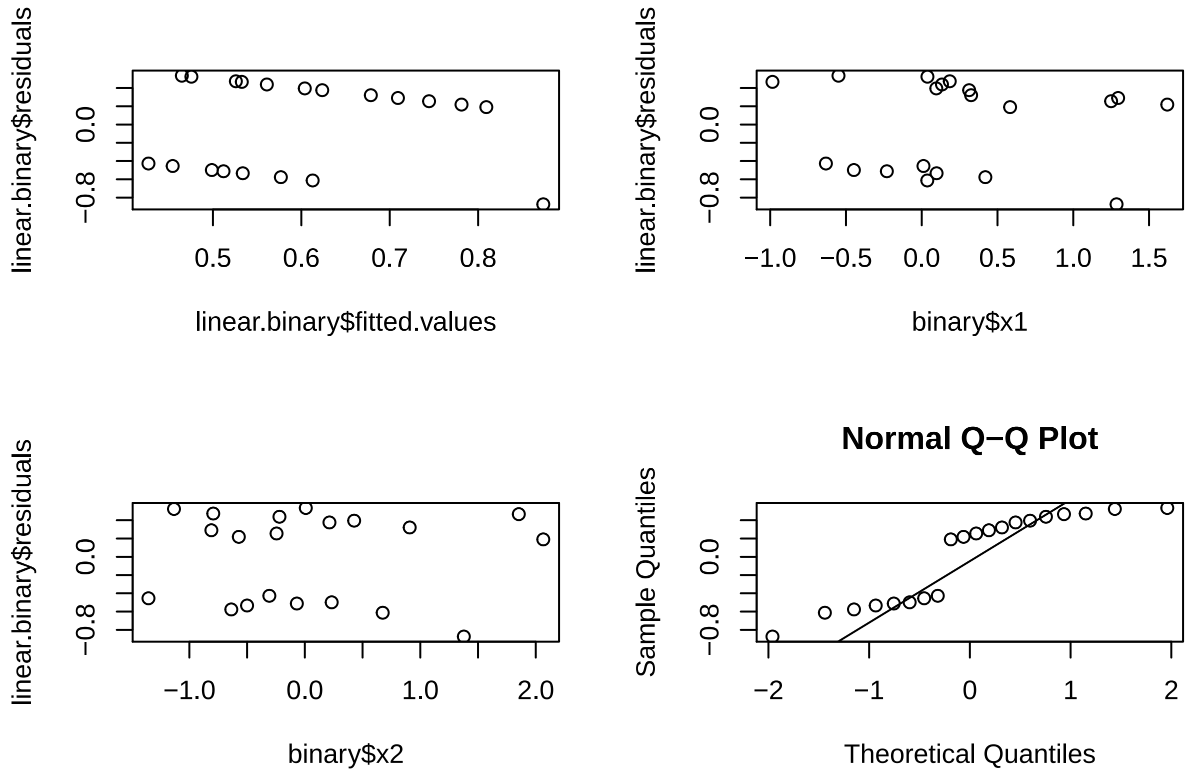
Linear Regression

```
#linear regression
linear.binary <- glm(y ~., family = "gaussian", data = binary)
summary(linear.binary)
```

```
##
## Call:
## glm(formula = y ~ ., family = "gaussian", data = binary)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8737  -0.5021   0.2371   0.4068   0.5352
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.55548    0.12271   4.527 0.000298 ***
## x1           0.16614    0.17184   0.967 0.347186
## x2           0.07599    0.12665   0.600 0.556431
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.2640822)
##
##     Null deviance: 4.8000  on 19  degrees of freedom
## Residual deviance: 4.4894  on 17  degrees of freedom
## AIC: 34.877
##
## Number of Fisher Scoring iterations: 2
```

Diagnosis of Linear Regression Model Fit

```
par(mfrow = c(2, 2))
plot(linear.binary$fitted.values, linear.binary$residuals)
plot(binary$x1, linear.binary$residuals)
plot(binary$x2, linear.binary$residuals)
qqnorm(linear.binary$residuals)
qqline(linear.binary$residuals)
```



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

When I examine the scatter plot of residuals and fitted values, I see that the scatter is not random. There are two distinct lines, which suggest that the OLS model does not fit the data well. When I examine the scatter plots of residuals and the predictors $x1$ and $x2$, I see more of equal variation and random scatter in the data.

The probability plot of residuals reveal that the errors seem to be approximately normally distributed.

Logistic Regression

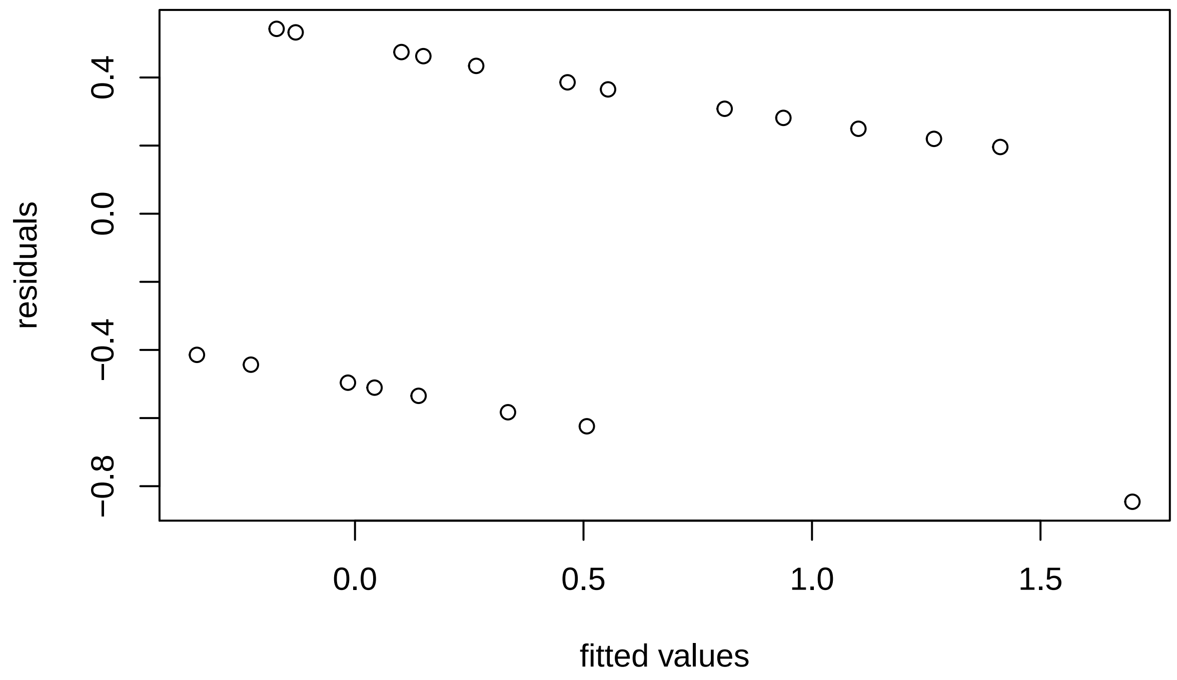
```
logistic.binary <- glm(y ~., family = "binomial", data = binary)
summary(logistic.binary)
```

```
##
## Call:
## glm(formula = y ~ ., family = "binomial", data = binary)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.933  -1.177   0.731   1.007   1.251
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.2415     0.4873   0.496    0.620
## x1            0.7573     0.7552   1.003    0.316
## x2            0.3530     0.5485   0.643    0.520
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 26.920  on 19  degrees of freedom
## Residual deviance: 25.557  on 17  degrees of freedom
## AIC: 31.557
##
## Number of Fisher Scoring iterations: 4
```

Diagnosis of Logistic Regression Model Fit

To diagnose the fit of the logistic regression model, I will plot deviance residuals against the fitted values as well as to the two predictors. If the model fits the data well, then I should see an even variation in the deviance residuals.

```
library(faraway)
linpred <- predict(logistic.binary)
predprob <- predict(logistic.binary, type="response")
rawres <- binary$y - predprob
plot(rawres ~ linpred, xlab="fitted values", ylab="residuals")
```



Based on the plot, I see an even variation as the fitted values vary, so this plot reveals no inadequacy in the model.

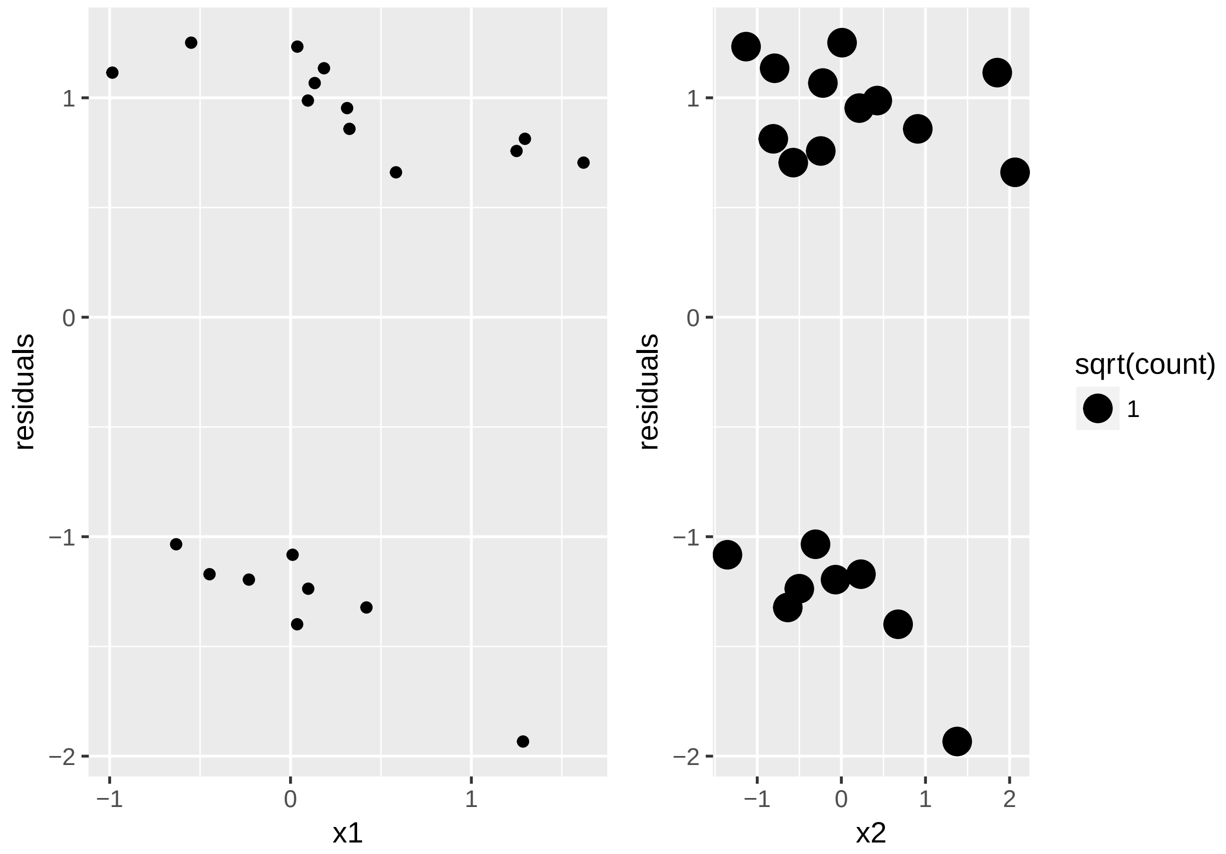
```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
binary <- mutate(binary, residuals=residuals(logistic.binary), linpred=predict(logistic.binary))
gdf <- group_by(binary, cut(linpred, breaks=unique(quantile(linpred, (1:7)/8))))
diagdf <- summarise(gdf, residuals=mean(residuals), linpred=mean(linpred))
gdf <- group_by(binary, x1)
diagdf <- summarise(gdf, residuals=mean(residuals))
library(ggplot2)
p1 <- ggplot(diagdf, aes(x=x1,y=residuals)) +
  geom_point()
p2 <- group_by(binary, x2) %>%
  summarise(residuals=mean(residuals), count=n()) %>%
  ggplot(aes(x=x2, y=residuals, size=sqrt(count))) +
  geom_point()
```

```r
source("multiplot.R")
multiplot(p1, p2, cols = 2)
```

```
## Loading required package: grid
```



The plots of deviance residuals vs the predictors $x1$ and $x2$ also reveal constant variation, so the model reveals no inadequacy.

Probit Regression

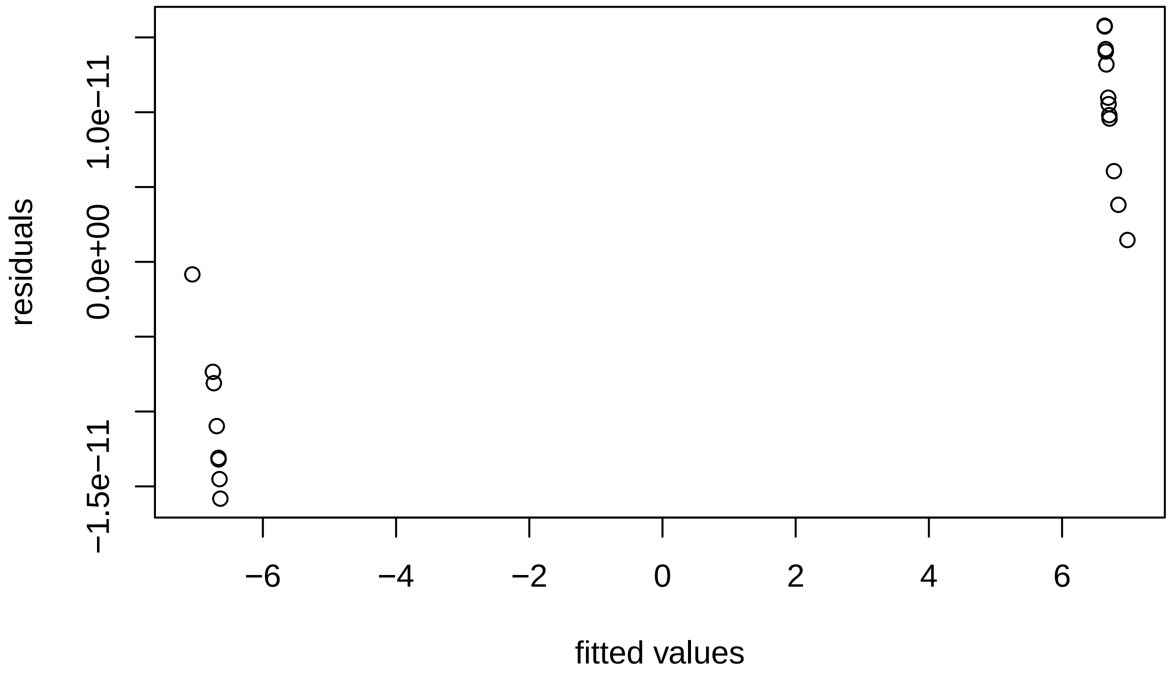
```r
probit.binary <- glm(y ~., family = binomial(link=probit), data = binary)
summary(probit.binary)
```

```
##
## Call:
## glm(formula = y ~ ., family = binomial(link = probit), data = binary)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -5.625e-06  -4.191e-06   3.122e-06   4.797e-06   5.617e-06
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 5.515e-01  6.385e+03   0.000    1.000
## x1          1.713e+00  1.210e+04   0.000    1.000
## x2          8.201e-01  8.729e+03   0.000    1.000
## residuals   5.660e+00  5.984e+03   0.001    0.999
## linpred            NA         NA      NA       NA
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2.6920e+01  on 19  degrees of freedom
## Residual deviance: 4.1818e-10  on 16  degrees of freedom
## AIC: 8
##
## Number of Fisher Scoring iterations: 23
```

Diagnosis of Probit Regression Model Fit

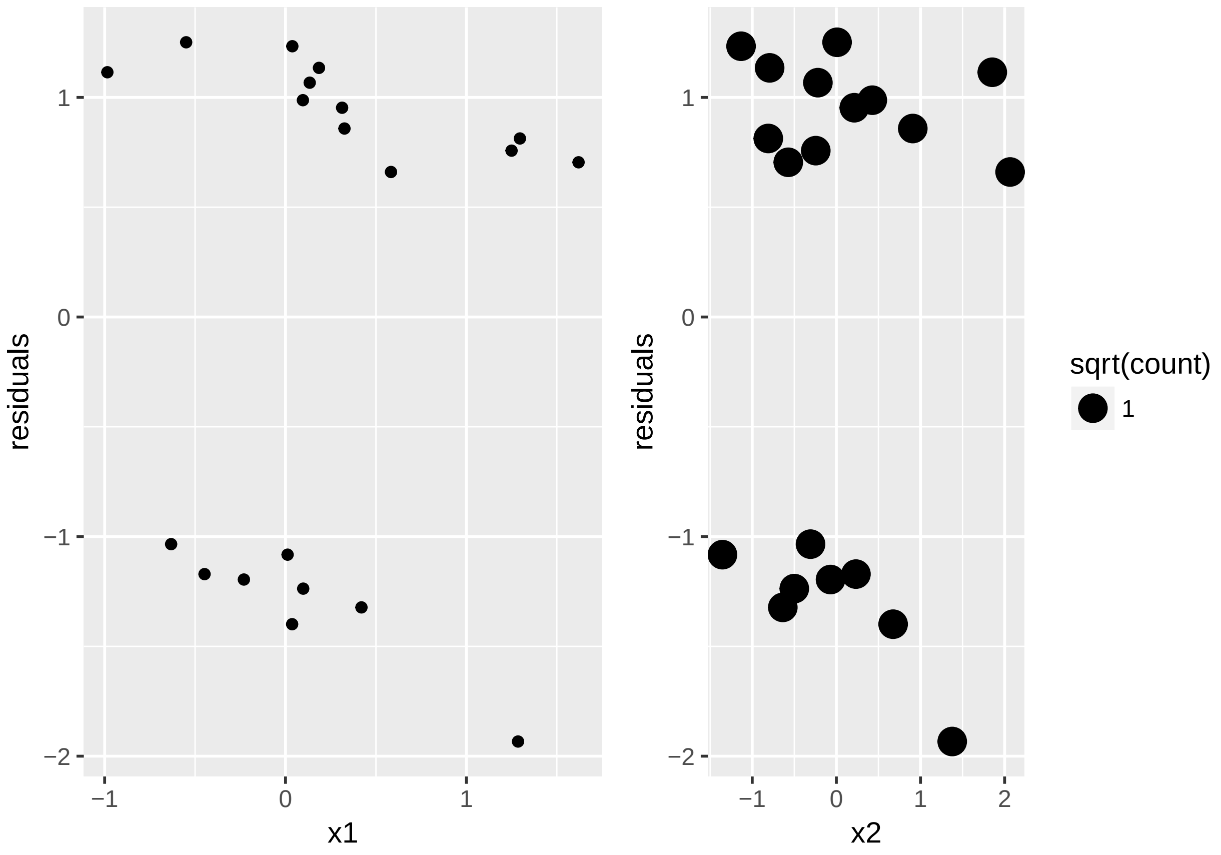
To diagnose the fit of the probit regression model, I will plot deviance residuals against the fitted values as well as to the two predictors. If the model fits the data well, then I should see an even variation in the deviance residuals.

```
linpred <- predict(probit.binary)
predprob <- predict(probit.binary, type="response")
rawres <- binary$y - predprob
plot(rawres ~ linpred, xlab="fitted values", ylab="residuals")
```



Based on the plot, I see an even variation as the fitted values vary, so this plot reveals no inadequacy in the model.

```r
binary <- mutate(binary, residuals=residuals(logistic.binary), linpred=predict(logistic.binary))
gdf <- group_by(binary, cut(linpred, breaks=unique(quantile(linpred, (1:7)/8))))
diagdf <- summarise(gdf, residuals=mean(residuals), linpred=mean(linpred))
gdf <- group_by(binary, x1)
diagdf <- summarise(gdf, residuals=mean(residuals))
p1 <- ggplot(diagdf, aes(x=x1,y=residuals)) +
  geom_point()
p2 <- group_by(binary, x2) %>%
  summarise(residuals=mean(residuals), count=n()) %>%
  ggplot(aes(x=x2, y=residuals, size=sqrt(count))) +
  geom_point()
multiplot(p1, p2, cols = 2)
```



The plots of deviance residuals vs the predictors $x1$ and $x2$ also reveal constant variation, so the model reveals no inadequacy.

The estimates and the standard errors of the three methods vary. The linear model has the smallest standard errors for the estimates of the coefficients. The logistic model had the largest standard errors. However, in terms of AIC, logistic regression has the lowest AIC value, which means that logistic regression fits the data best among the three methods.

Logistic regression is often preferred over probit regression because logistic regression leads to simpler mathematics than the probit due to the intractability of Φ. Also, logistic regression is easier to interpret using odds.

2.2 The *sale* dataset contains the counts of two types of products over 100 days. Do the following steps:

- Linear Regression
- Poisson Regression
- Negative binomial regression
- Zero inflated Poisson regression

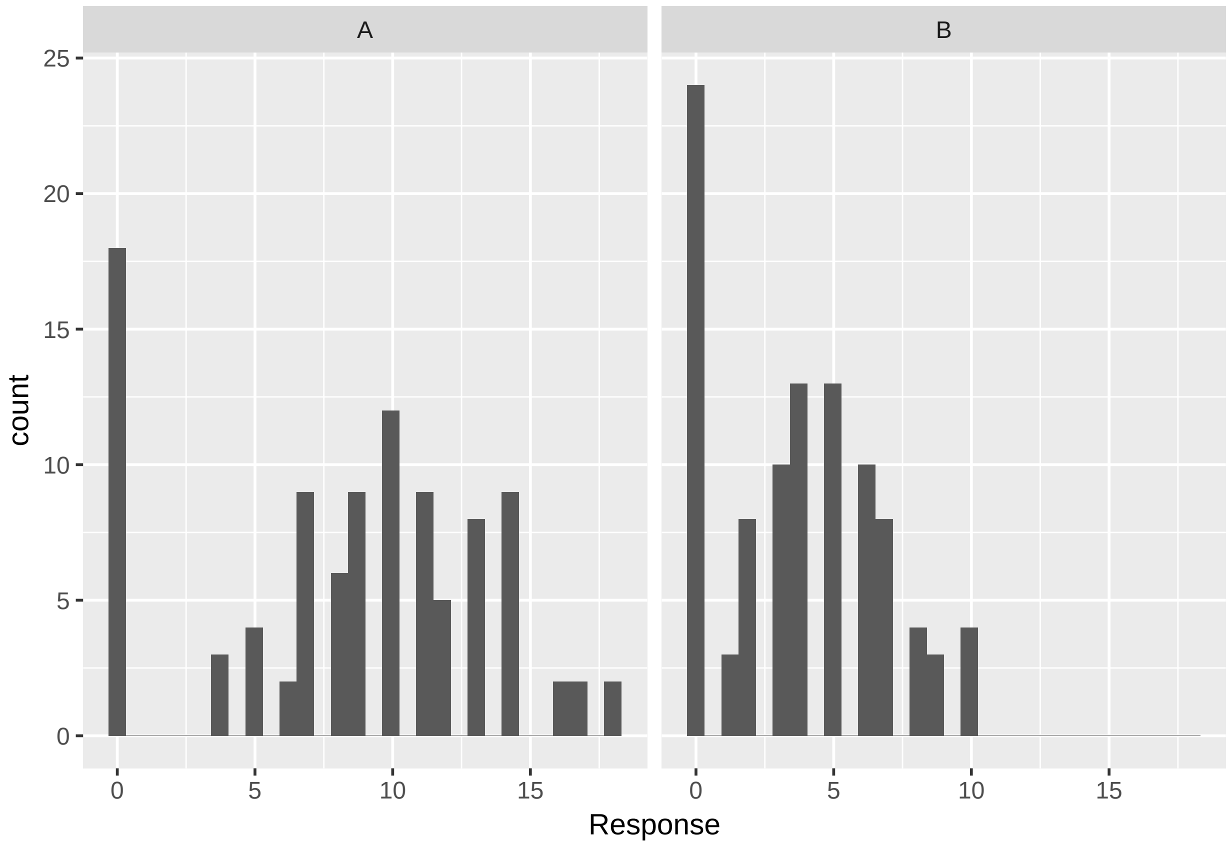
Compare the results. Briefly describe the advantages and the disadvantages of these methods.

```r
load("sale.RData")
head(data.zip)
```

```
##   Trt Response
## 1   A       13
## 2   A        0
## 3   A       14
## 4   A       18
## 5   A       14
## 6   A       13
```

```r
ggplot(data = data.zip, mapping = aes(x = Response)) +
  geom_histogram() +
  facet_wrap(~ Trt)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

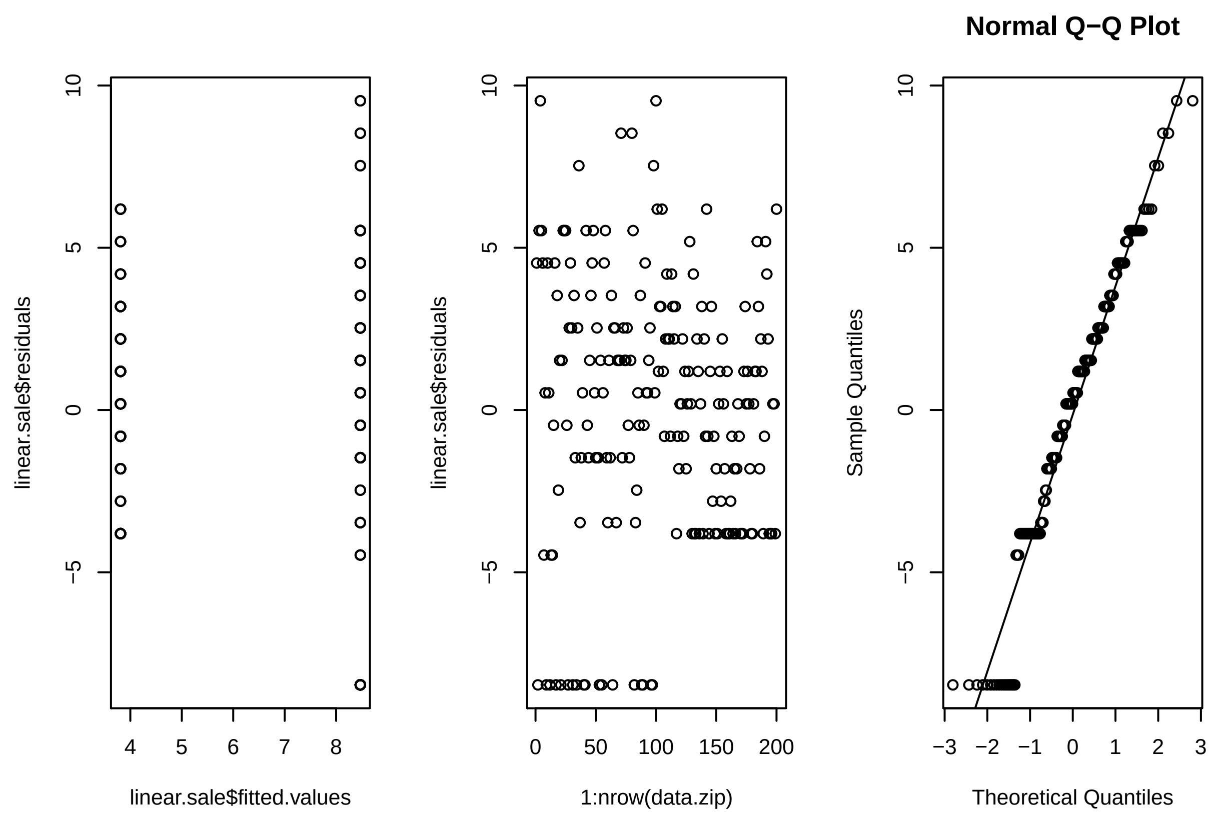


The data is count data with a point mass at Response = 0. Based on this, zero inflated Poisson regression would fit the data best. But we need to watch out for overdispersion.

```
#linear regression
linear.sale <- glm(Response ~., family = "gaussian", data = data.zip)
summary(linear.sale)
```

```
##
## Call:
## glm(formula = Response ~ ., family = "gaussian", data = data.zip)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
##  -8.47   -2.81    0.36    2.53    9.53
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.4700     0.4095  20.684  < 2e-16 ***
## TrtB         -4.6600     0.5791  -8.047 7.69e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 16.76919)
##
##     Null deviance: 4406.1  on 199  degrees of freedom
## Residual deviance: 3320.3  on 198  degrees of freedom
## AIC: 1135.5
##
## Number of Fisher Scoring iterations: 2
```

```
par(mfrow = c(1, 3))
plot(x = linear.sale$fitted.values, y = linear.sale$residuals)
plot(x = 1:nrow(data.zip), y = linear.sale$residuals)
qqnorm(linear.sale$residuals)
qqline(linear.sale$residuals)
```



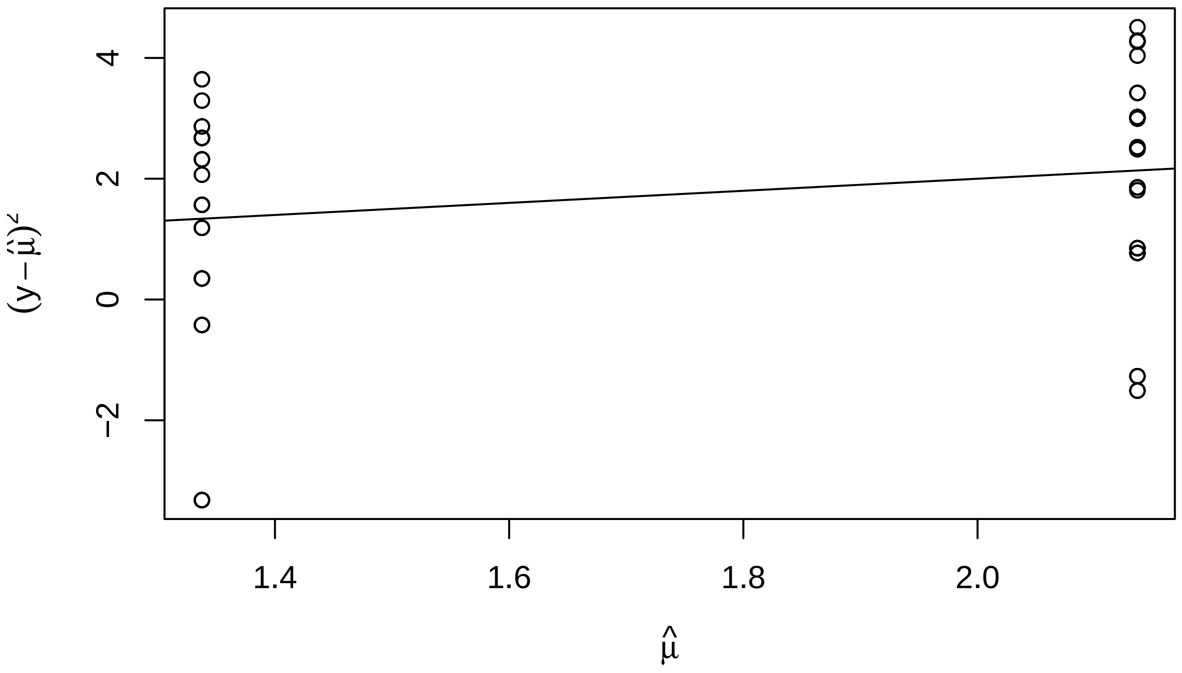
```r
par(mfrow = c(1, 1))
```

Based on the diagnostic plot of residuals vs the fitted values, I see that the linear model does not fit the data well because the residuals do not have random scatter. Also, the spread does not seem to be constant.

```r
#Poisson regression
Poisson.sale <- glm(Response ~., family = poisson(link = "log"), data = data.zip)
summary(Poisson.sale)
```

```
##
## Call:
## glm(formula = Response ~ ., family = poisson(link = "log"), data = data.zip)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.1158  -1.2919   0.1384   1.0342   2.8422
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.13653    0.03436   62.18   <2e-16 ***
## TrtB        -0.79890    0.06169  -12.95   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 896.95  on 199  degrees of freedom
## Residual deviance: 715.60  on 198  degrees of freedom
## AIC: 1314.8
##
## Number of Fisher Scoring iterations: 5
```

```
plot(log(fitted(Poisson.sale)),
     log((data.zip$Response - fitted(Poisson.sale))^2),
     xlab=expression(hat(mu)),
     ylab=expression((y-hat(mu))^2))
abline(0,1)
```



Based on the plot of mean vs variance, the Poisson model does not fit the data well because of the point mass at zero. The variance does not seem to equal the mean.

```
#negative binomial regression
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```
negbin.sale <- glm.nb(Response ~., data = data.zip)
summary(negbin.sale)
```

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

```
## glm.nb(formula = Response ~ ., data = data.zip, init.theta = 1.767750023,
##     link = log)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.49192  -0.61506   0.06416   0.55162   1.31455
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.13653    0.08269  25.838  < 2e-16 ***
## TrtB        -0.79890    0.12296  -6.497 8.18e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(1.7678) family taken to be 1)
##
##     Null deviance: 300.90  on 199  degrees of freedom
## Residual deviance: 258.92  on 198  degrees of freedom
## AIC: 1123.5
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  1.768
##           Std. Err.:  0.289
##
##  2 x log-likelihood:  -1117.540
```

```
#zero inflated possion regression
library(pscl)
```

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

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:faraway':
##
##     melanoma
```

```
## Classes and Methods for R developed in the
```

```
## Political Science Computational Laboratory
```

```
## Department of Political Science
```

```
## Stanford University
```

```
## Simon Jackman
```

```
## hurdle and zeroinfl functions by Achim Zeileis
```

```
zeroinfl.sale <- zeroinfl(Response ~., data = data.zip)
summary(zeroinfl.sale)
```

```
##
## Call:
## zeroinfl(formula = Response ~ ., data = data.zip)
##
```

```
## Pearson residuals:
##      Min       1Q   Median       3Q      Max 
## -1.72122 -0.70515  0.08691  0.76190  2.15346 
## 
## Count model coefficients (poisson with log link):
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  2.33495    0.03437   67.94   <2e-16 ***
## TrtB        -0.72980    0.06245  -11.69   <2e-16 ***
## 
## Zero-inflation model coefficients (binomial with logit link):
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  -1.5165     0.2603  -5.825  5.7e-09 ***
## TrtB          0.3348     0.3538   0.946    0.344    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Number of iterations in BFGS optimization: 12 
## Log-likelihood: -484.5 on 4 Df
```

```
#aic for zero-inflated Poisson regression
aic.zeroinfl <- -2 * zeroinfl.sale$loglik + 2 * 4
aic.zeroinfl
```

```
## [1] 976.9171
```

Based on the results of the four models, zero-inflated Poisson regression performed the best since it had the lowest AIC value. Linear regression model had an estimate of the effect of treatment on response that is quite a bit different from the rest of the models. The other three methods had similar estimated effects. All models indicate a negative effect. Poisson regression and zero-inflated Poisson regression had the smallest standard errors of the estimates. However, based on the plot of mean and variance, I see that the Poisson regression does not fit the model well.

I will also perform goodness of fit tests for Poisson and negative binomial regressions.

```
pchisq(deviance(Poisson.sale),df.residual(Poisson.sale),lower=FALSE)
```

```
## [1] 1.070159e-59
```

```
pchisq(deviance(negbin.sale),df.residual(negbin.sale),lower=FALSE)
```

```
## [1] 0.002351528
```

Here, I can see that the p-values are statistically significant, so I reject the null hypothesis that the models fit the data well. Negative binomial regression fits the model better as expected due to overdispersion. However, it's still not fitting the model well due to the point mass at zero.

The advantage of linear regression is that when the model assumptions are met and when the outcome is continuous, the linear regression model is BLUE. The disadvantage is that the assumptions required are very restrictive, so many datasets cannot be modeled by OLS.

The advantage of Poisson regression is that it can be applied to model count data. The disadvantage is that it requires the assumption that the mean is equal to the variance. Another disadvantage is that sometimes we see count response data where the number of zeroes appearing is significantly greater than the Poisson regression model would predict.

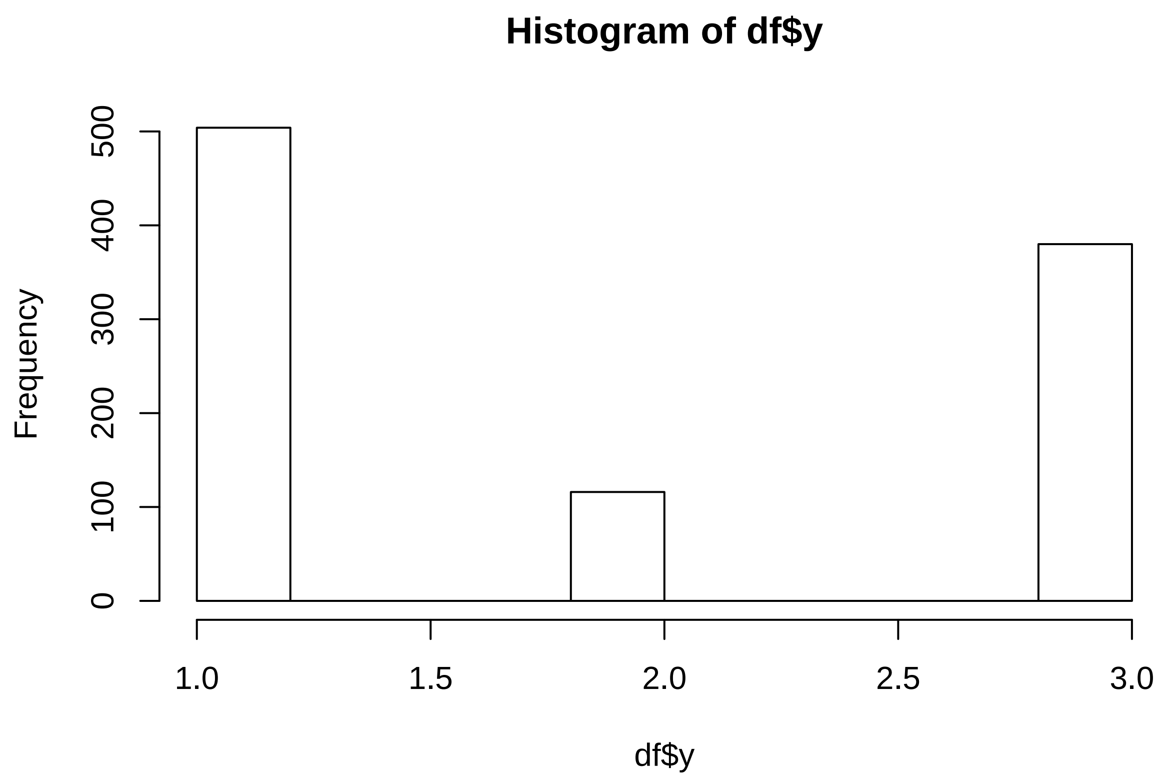
The advantage of negative binomial regression is that it can be applied to model count data even when overdispersion exists. The disadvantage is that sometimes we see count response data where the number of zeroes appearing is significantly greater than the negative binomial regression model would predict.

The advantage of zero-inflated Poisson regression is that it can model data that has a point mass at zero.

The disadvantage is that the model still assumes that the mean and the variance are equal for the non-zero part of the data.

3. The *multilevel* dataset contains the response with multiple levels. Fit a multinomial regression on this dataset and report the result.

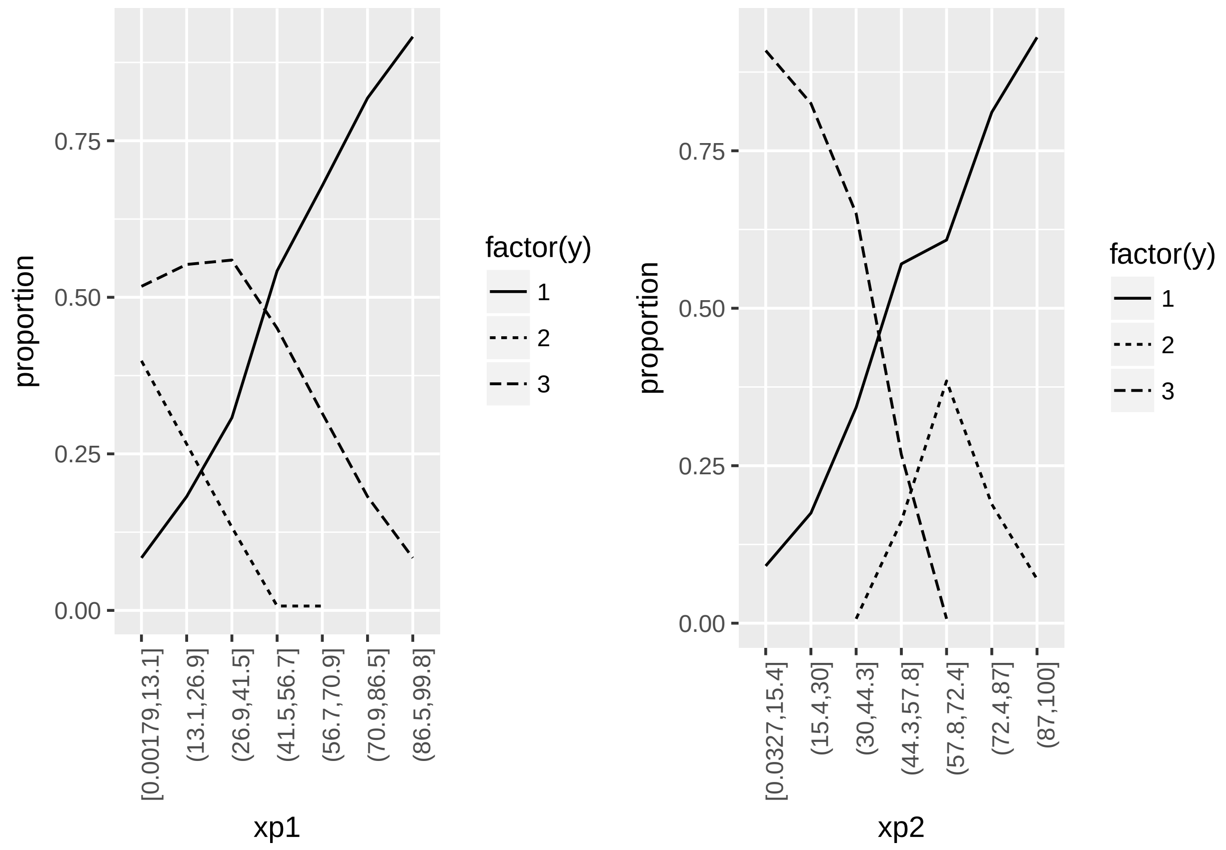
```r
load("multilevel.RData")
hist(df$y)
```



Based on the histogram, I see that the response variable is categorical.

```r
x1p <- mutate(df, xp1 =cut_number(x1,7)) %>%
  group_by(xp1, y) %>%
  summarise(count=n()) %>%
  group_by(xp1) %>%
  mutate(total=sum(count), proportion=count/total)
p1 <- ggplot(x1p, aes(x=xp1, y=proportion, group=y, linetype = factor(y)))+
  geom_line() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

x2p <- mutate(df, xp2 =cut_number(x2,7)) %>%
  group_by(xp2, y) %>%
  summarise(count=n()) %>%
  group_by(xp2) %>%
  mutate(total=sum(count), proportion=count/total)
```

```r
p2 <- ggplot(x2p, aes(x=xp2, y=proportion, group=y, linetype = factor(y)))+
  geom_line() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
multiplot(p1, p2, cols = 2)
```



The plots above show the following:

As $x1$ increases, $P(y = 1)$ increases while $P(y = 2)$ and $P(y = 3)$ decrease.

As $x2$ increases, $P(y = 1)$ increases, $P(y = 2)$ increases and then decreases, and $P(y = 3)$ decreases.

Now I fit a multinomial regression to the data.

```r
library(nnet)
multilevel.df <- multinom(y ~., data = df[, 1:3])
```

```
## # weights:  12 (6 variable)
## initial  value 1098.612289
## iter  10 value 243.883459
## iter  20 value 218.982692
## final  value 218.979741
## converged
```

```r
summary(multilevel.df)
```

```
## Call:
## multinom(formula = y ~ ., data = df[, 1:3])
##
## Coefficients:
```

```
##   (Intercept)         x1         x2
## 2    14.43495 -0.2085226 -0.1270780
## 3    23.87802 -0.2177823 -0.3012766
##
## Std. Errors:
##   (Intercept)         x1         x2
## 2    1.728311 0.01971651 0.01782759
## 3    1.980376 0.01909514 0.02417414
##
## Residual Deviance: 437.9595
## AIC: 449.9595
```

The intercept terms model the probabilities of y when $x1$ and $x2$ are zero. The slope terms represent the log-odds of moving from the baseline category of 1 to categories 2 and 3, respectively, for a unit change of 1 in $x1$ and $x2$. Negative coefficients mean that the log-odds of $y = 1$ to $y = 2$ and $y = 3$ decrease as $x1$ and $x2$ increase.

4. The *cheese* dataset has four brands of cheese. Their brand names are just their rates, "A", "B", "C", and "D". Fit both the proportional odds model and the ordered probit model on this dataset. Compare the results.

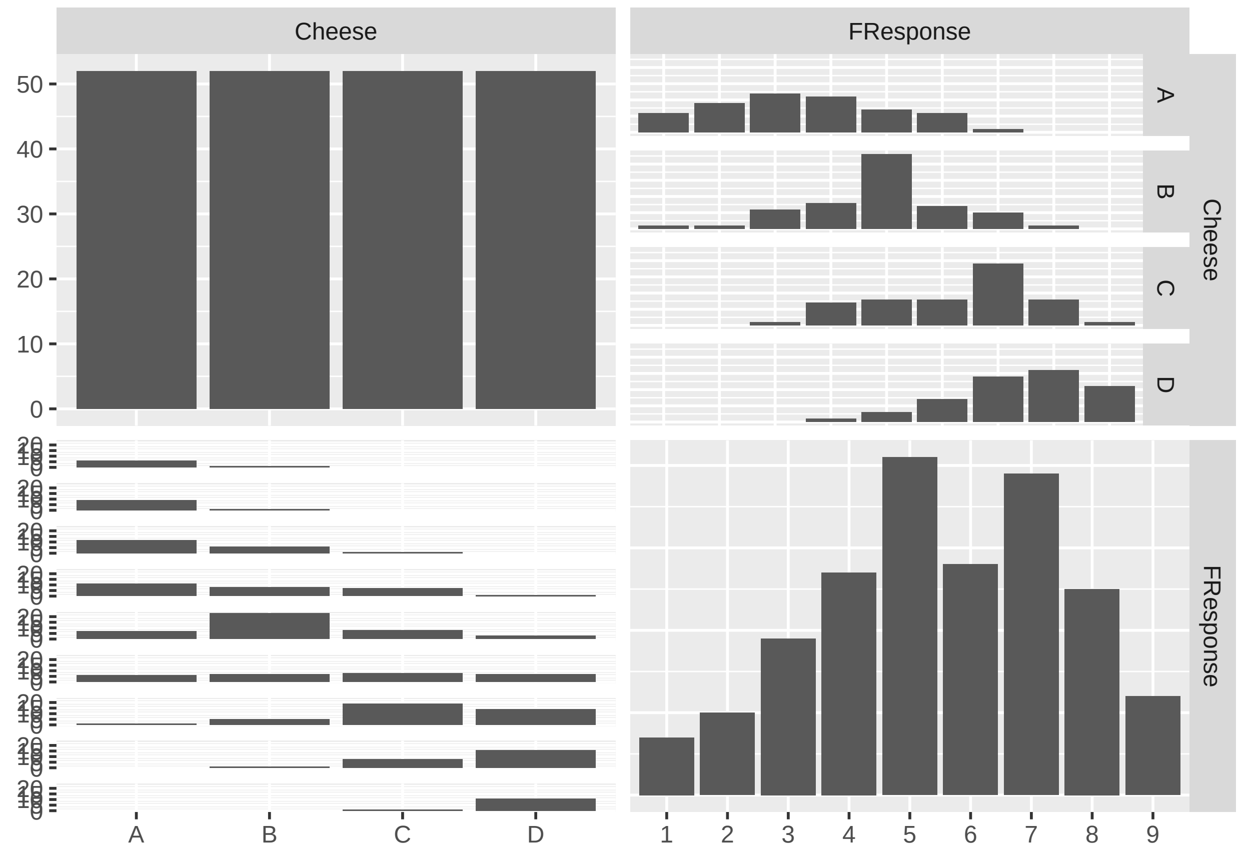
```r
load("cheese.RData")
str(cheese)
```

```
## 'data.frame':    208 obs. of  3 variables:
##  $ Cheese   : Factor w/ 4 levels "A","B","C","D": 3 3 3 3 3 3 3 3 3 3 ...
##  $ Response : int  3 4 4 4 4 4 4 4 5 5 ...
##  $ FResponse: Ord.factor w/ 9 levels "1"<"2"<"3"<"4"<..: 3 4 4 4 4 4 4 4 5 5 ...
```

```r
library(GGally)
```

```
##
## Attaching package: 'GGally'

## The following object is masked from 'package:dplyr':
##
##     nasa

## The following object is masked from 'package:faraway':
##
##     happy
```

```r
ggpairs(cheese[,-2])
```



Proportional Odds Model

```
prop.odds.cheese <- polr(FResponse ~ Cheese, data = cheese)
prop.odds.cheese$coefficients[1]
```

```
##  CheeseB
## 1.641975
```

```
summary(prop.odds.cheese)
```

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

```
## Call:
## polr(formula = FResponse ~ Cheese, data = cheese)
##
## Coefficients:
##         Value Std. Error t value
## CheeseB 1.642     0.3753   4.375
## CheeseC 3.352     0.4287   7.819
## CheeseD 4.965     0.4767  10.414
##
## Intercepts:
##     Value   Std. Error t value
## 1|2 -2.1155  0.4106    -5.1525
## 2|3 -1.0603  0.3009    -3.5243
## 3|4  0.0392  0.2735     0.1433
## 4|5  1.1079  0.2969     3.7309
## 5|6  2.4441  0.3397     7.1940
## 6|7  3.3961  0.3724     9.1192
## 7|8  4.8978  0.4249    11.5281
## 8|9  6.4576  0.5045    12.7993
##
## Residual Deviance: 711.3479
## AIC: 733.3479
```

```
#Probabilities at cheese A (baseline probabilities)
cheeseAprob <- c(ilogit(prop.odds.cheese$zeta[1]),
         ilogit(prop.odds.cheese$zeta[2:8]) - ilogit(prop.odds.cheese$zeta[1:7]),
         1 - ilogit(prop.odds.cheese$zeta)[8])
cheeseAprob
```

```
##         1|2         2|3         3|4         4|5         5|6         6|7
## 0.107596784 0.149648422 0.252552765 0.241930442 0.168400703 0.047454140
##         7|8         8|9         8|9
## 0.025008745 0.005841894 0.001566104
```

```
sum(cheeseAprob)
```

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

The results from the proportional odds model state the following:

- The odds of moving from a lower response to a higher response by one level increase by a factor of $exp(1.6419747)$ or 5.1653592 as cheese changes from A to B.
- The odds of moving from a lower response to a higher response by one level increase by a factor of $exp(3.3518749)$ or 28.556224 as cheese changes from A to C.
- The odds of moving from a lower response to a higher response by one level increase by a factor of $exp(4.9646488)$ or 143.2582288 as cheese changes from A to D.

Since the log-odds are similar to those obtained in the multinomial logit model, the intercepts can be used the predict probability of given a particular response when cheese group is A. So the probabilities of giving responses 1 or 2 or 3 or 4 or 5 or 6 or 7 or 8 or 9 when cheese group is A are 0.1075968, 0.1496484, 0.2525528, 0.2419304, 0.1684007, 0.0474541, 0.0250087, 0.0058419, 0.0015661, respectively.

Ordered Probit Model

```
ordered.probit.cheese <- polr(FResponse ~ Cheese, method = "probit", data = cheese)
summary(ordered.probit.cheese)
```

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

```
## Call:
## polr(formula = FResponse ~ Cheese, data = cheese, method = "probit")
##
## Coefficients:
##         Value Std. Error t value
## CheeseB 0.921     0.2101   4.383
## CheeseC 1.898     0.2267   8.371
## CheeseD 2.862     0.2496  11.468
##
## Intercepts:
##     Value   Std. Error t value
## 1|2 -1.2143  0.2080    -5.8374
## 2|3 -0.6468  0.1718    -3.7659
## 3|4 -0.0010  0.1616    -0.0060
## 4|5  0.6262  0.1674     3.7401
## 5|6  1.3977  0.1808     7.7305
## 6|7  1.9464  0.1950     9.9821
## 7|8  2.8342  0.2227    12.7255
## 8|9  3.7398  0.2633    14.2013
##
## Residual Deviance: 707.3907
## AIC: 729.3907
```

The deviance and the AIC between the proportional odds model and the ordered probit model are similar. The ordered probit model has slightly smaller deviance and AIC, which indicates that it fits the model better (though only slightly). The coefficient estimates appear to be different but this is due to the inverse CDF of standard normal function.