An Introduction to Categorical Data Analysis

2019-08-14

Contents

PREFACE

This document is the R implementation of the examples and exercises in the textbook. An Introduction to Categorical Data Analysis, Second Edition (https://onlinelibrary.wiley.com/doi/book/10.1002/0470114754)

Here are some instructions for this document:

1. The documentation is accompanied by R packages `cdabookdb` and `cdabookfunc`. These R packages contain the data used in the textbook and some of the functions used. Please see Appendix A for instructions on how to install and use this package.
2. The datasets referenced in each case can be found in the `cdabookdb` package. All the data used in the case in the document and the data of all the exercises in the textbook can be found in the list of data set names in the package Appendix B.
3. Each case in the document is independent, which means that the results of the later cases will not take advantage of the previously calculated results or loaded packages.
4. The chapter numberand also the chapter titles in the document are consistent with the textbook. Therefore, if there is no need for R implementation in the corresponding chapter of the textbook, the content of this chapter in this document is empty.
5. The document is completed by several people, and the code style and description style will be different.
6. The document has now completed the R implementation of the first eight chapters of the case, the `cdabookdb` package has now completed the entry of all the data in the first eight chapters.
7. The documentation is available in a variety of formats and can be downloaded from the download button at the top of the web version (gitbook version). They are pdf version, equb3 version, zip version (compressed version of gitbook version).

Chapter 1

INTRODUCTION

1.1 Categorical Response Data

1.2 Distributions for Categorical Data

Binomial Distribution

```r
# Calculation for binomial distribution
dbinom(0, 10, 0.2)  # 10 trials, possibility of success:0.2, succeed 0.
```

```
## [1] 0.1074
```

```r
# CDF with given parameters
n <- 10
prob_matrix <- sapply(c(0.2, 0.5, 0.8), function(p) pbinom(0:n, n, p))
dimnames(prob_matrix) <- list(0:n, c("P=0.2", "P=0.5", "P=0.8"))
xtable::xtable(prob_matrix, align = "cccc", digits = 3)
```

P=0.2	P=0.5	P=0.8
0.107	0.001	0.000
0.376	0.011	0.000
0.678	0.055	0.000
0.879	0.172	0.001
0.967	0.377	0.006
0.994	0.623	0.033
0.999	0.828	0.121
1.000	0.945	0.322
1.000	0.989	0.624
1.000	0.999	0.893
1.000	1.000	1.000

```
# Mean and standard deviation of binomial distribution with given parameters
n <- 10
p <- 0.2
n * p  # Mean
```

```
## [1] 2
```

```
sqrt(n * p * (1 - p))  # Standard deviation
```

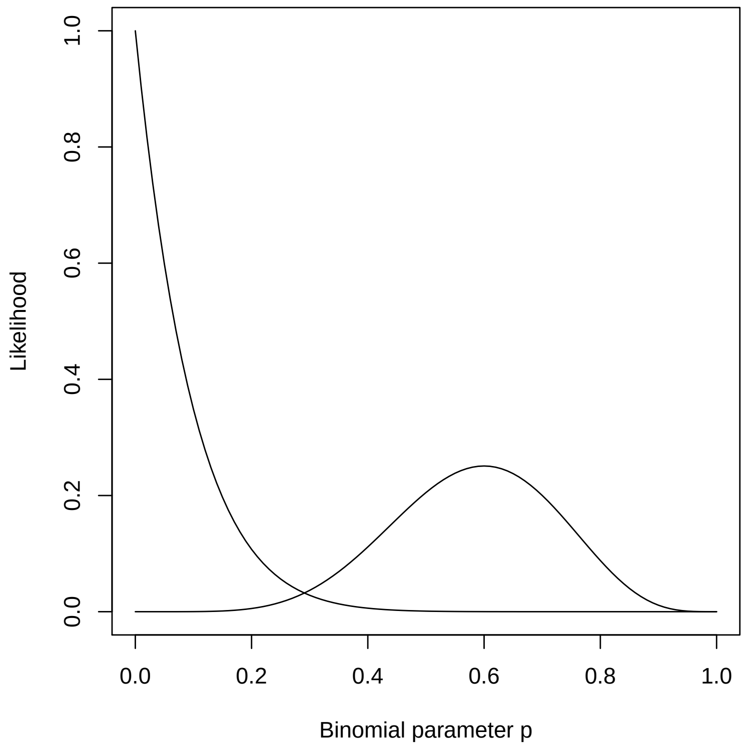
```
## [1] 1.265
```

1.3 Statistical Inference for Categorical Data

Likelihood of Binomial Distribution

Figure 1.1 in the book.

```
prob <- seq(0, 1, 0.01)
prob_plot_data <- data.frame(
  Prob = prob,
  Y_0 = dbinom(0, 10, prob),
  Y_6 = dbinom(6, 10, prob)
)

par(pty = "s")
plot(
  Y_0 ~ Prob, type = "l",
  data = prob_plot_data,
  asp = 1,
  xlab = "Binomial parameter  ",
  ylab = "Likelihood"
)
lines(Y_6 ~ Prob, type = "l", data = prob_plot_data)
```



Significance Test About a Binomial Parameter

The binomial distribution test is divided into two types. The data used in the following examples are from the 1.3.3 section of the legalization of abortion survey.

One is the exact binomial test, use`binom.test()`

```
binom.test(400, 893)
```

```
##
##  Exact binomial test
##
## data:  400 and 893
## number of successes = 400, number of trials = 893,
## p-value = 0.002
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
```

```
##  0.4150 0.4812
## sample estimates:
## probability of success
##                 0.4479
```

Another is the approximate binomial test of normal (or chi-square), use `prob.test()`

```
prop.test(400, 893)
```

```
##
##  1-sample proportions test with continuity correction
##
## data:  400 out of 893, null probability 0.5
## X-squared = 9.5, df = 1, p-value = 0.002
## alternative hypothesis: true p is not equal to 0.5
## 95 percent confidence interval:
##  0.4151 0.4813
## sample estimates:
##      p
## 0.4479
```

```
# correst=FALSE means it's without continuity adjustment
prop.test(400, 893, correct = FALSE)
```

```
##
##  1-sample proportions test without continuity
##  correction
##
## data:  400 out of 893, null probability 0.5
## X-squared = 9.7, df = 1, p-value = 0.002
## alternative hypothesis: true p is not equal to 0.5
## 95 percent confidence interval:
##  0.4156 0.4807
## sample estimates:
##      p
## 0.4479
```

Sections 1.3.2 and 1.3.3 introduce and use large sample approximation without continuity adjustment.

The P-values of the three tests are all less than 0.05, thus rejecting the original hypothesis.

Confidence Intervals for a Binomial Parameter

Confidence intervals are included in the output of the previous section [Binomial Distribution Hypothesis Test] (# binom-test)

Among them, `prop. test (correct = FALSE)` outputs confidence intervals calculated by the first adjustment method introduced in the book

```
prop.test(9, 10, 0.9, correct = FALSE)$conf.int
```

```
## Warning in prop.test(9, 10, 0.9, correct = FALSE): Chi-
## squared approximation may be incorrect
```

```
## [1] 0.5958 0.9821
## attr(,"conf.level")
## [1] 0.95
```

For the second adjustment method, Agresti–Coull confidence interval, R has no built-in function to calculate,but it can be calculated by the `binom. agresti. Coull()' function in the`binom `package. (At the same time, the`binom. confint()‘ function can also be used to calculate the summary table of various confidence intervals.)

```
library(binom)
binom.agresti.coull(9, 10)
```

```
##          method x  n mean lower upper
## 1 agresti-coull 9 10  0.9 0.574 1.004
```

```
binom.confint(9, 10)
```

```
##           method x  n   mean  lower  upper
## 1  agresti-coull 9 10 0.9000 0.5740 1.0039
## 2     asymptotic 9 10 0.9000 0.7141 1.0859
## 3          bayes 9 10 0.8636 0.6692 0.9996
## 4        cloglog 9 10 0.9000 0.4730 0.9853
## 5          exact 9 10 0.9000 0.5550 0.9975
## 6          logit 9 10 0.9000 0.5328 0.9861
## 7         probit 9 10 0.9000 0.5879 0.9904
## 8        profile 9 10 0.9000 0.6283 0.9904
## 9            lrt 9 10 0.9000 0.6284 0.9940
## 10     prop.test 9 10 0.9000 0.5412 0.9948
## 11        wilson 9 10 0.9000 0.5958 0.9821
```

1.4 Statistical Inference for Discrete Data

Statistical Inference of Binomial Distribution Parameters

For Wald, Score and Likelihood-Ratio methods:

```
# set parameters
p <- 0.9
```

```r
n <- 10
pi <- 0.5
```

```r
# Wald test
SE <- sqrt(p * (1 - p) / n)
z <- (p - pi) / SE; z
```

```
## [1] 4.216
```

```r
# Score test
SE <- sqrt(pi * (1 - pi) / n)
z <- (p - pi) / SE; z
```

```
## [1] 2.53
```

```r
# likelihood-ratio test
x <- n * p
L0 <- dbinom(x, n, pi)
L1 <- dbinom(x, n, p)
z <- -2 * log(L0 / L1); z
```

```
## [1] 7.361
```

Or use `binom_inference()`function in `cdabookcode`:

```r
library(cdabookfunc)
binom_inference(0.9, 10, 0.5, method = "wald")
```

```
## $z
## [1] 4.216
##
## $method
## [1] "wald"
```

```r
binom_inference(0.9, 10, 0.5, method = "l")
```

```
## $z
## [1] 7.361
##
## $method
## [1] "likelihood-ratio test"
```

Small-Sample Binomial Inference

```r
# one-side test pvalue
# (H0: pi = 0.5) vs (H1: pi > 0.5)
# p-value = P(Y >= 9) = P(Y > 8)
```

```
1 - pbinom(8, 10, 0.5)
```

```
## [1] 0.01074
```

```
# two-side test pvalue
# (H0: pi = 0.5) vs (H1: pi != 0.5)
# p-value = 1 + P(Y <= 1) + P(Y >= 9) = 2 * P(Y > 8)
pbinom(1, 10, 0.5) + pbinom(8, 10, 0.5, lower.tail = FALSE)
```

```
## [1] 0.02148
```

```
2 * (1 - pbinom(8, 10, 0.5))
```

```
## [1] 0.02148
```

P-value adjustment

Small sample inference is conservative, we can use adjusted P-values.

Use `binom_mid_pvalue()` to calculate mid P-values.

```
library(cdabookfunc)
binom_mid_pvalue(9, 10, "g")  # right-tail p-value
```

```
## $pvalue
## [1] 0.005859
##
## $alternative
## [1] "greater"
```

```
binom_mid_pvalue(9, 10)  # two-sided p-value
```

```
## $pvalue
## [1] 0.01172
##
## $alternative
## [1] "two.sided"
```

```
# get table1.2
pvalue_matrix <- cbind(
  0:10,
  dbinom(0:10, 10, 0.5),
  1 - pbinom(-1:9, 10, 0.5),
  binom_mid_pvalue(0:10, 10, "g")$pvalue
)
dimnames(pvalue_matrix) <- list(0:10, c("y", "P(y)", "P-value", "Mid P-value"))
xtable::xtable(pvalue_matrix, align = "ccccc", digits = c(0, 0, 4, 4, 4))
```

y	P(y)	P-value	Mid P-value
0	0.0010	1.0000	0.9995
1	0.0098	0.9990	0.9941
2	0.0439	0.9893	0.9673
3	0.1172	0.9453	0.8867
4	0.2051	0.8281	0.7256
5	0.2461	0.6230	0.5000
6	0.2051	0.3770	0.2744
7	0.1172	0.1719	0.1133
8	0.0439	0.0547	0.0327
9	0.0098	0.0107	0.0059
10	0.0010	0.0010	0.0005

Chapter 2

CONTINGENCY TABLES

2.1 Probability Structure for Contingency Tables

Example: Belief in Afterlife

```r
library(cdabookdb)
data("afterlife1")
afterlife1
```

```
##          Belief
## Gender    Yes No or Undecided
##   Females 509             116
##   Males   398             104
```

```r
margin.table(afterlife1, margin = 1)  # calculate row sum
```

```
## Gender
## Females   Males
##     625     502
```

```r
margin.table(afterlife1, margin = 2)  # calculate column sum
```

```
## Belief
##              Yes No or Undecided
##              907             220
```

```r
addmargins(afterlife1)  # add sum of row and sum of column to contingency table
```

```
##          Belief
## Gender     Yes No or Undecided  Sum
##   Females  509             116  625
##   Males    398             104  502
##   Sum      907              220 1127
```

```
prop.table(afterlife1, margin = 1)  # conditional distribution with given row
```

```
##          Belief
## Gender       Yes No or Undecided
##   Females 0.8144          0.1856
##   Males   0.7928          0.2072
```

```
prop.table(afterlife1, margin = 2)  # conditional distribution with given column
```

```
##          Belief
## Gender       Yes No or Undecided
##   Females 0.5612          0.5273
##   Males   0.4388          0.4727
```

2.2 Comparing Proportions in Two-by-Two Tables

Example: Aspirin and Incidence of Heart Attacks

```
library(cdabookdb)
data("aspirin")
aspirin
```

```
##          MI
## Group         Y     N
##   Placebo   189 10845
##   Aspirin   104 10933
```

```
margin.table(aspirin, 1)  # the number of people who take placebo and Aspirin
```

```
## Group
## Placebo Aspirin
##   11034   11037
```

```
prop.table(aspirin, 1)  # the proportions of physicians that suffered MI in two groups
```

```
##          MI
## Group            Y        N
##   Placebo 0.017129 0.982871
##   Aspirin 0.009423 0.990577
```

```
prop.test(aspirin)  # test whether the two proportions are the same and calculate the confidence interva
```

```
##
##  2-sample test for equality of proportions with
##  continuity correction
```

```
##
## data:  aspirin
## X-squared = 24, df = 1, p-value = 8e-07
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  0.004597 0.010815
## sample estimates:
##   prop 1   prop 2
## 0.017129 0.009423
```

2.3 The Odds Ratio

Example: Odds Ratio for Aspirin Use and Heart Attacks

Odds ratio can be calculated using `odds ratio` in this document package `cdabook code`. For more information, please use `odds ratio`

```
library(cdabookfunc)
library(cdabookdb)
data("aspirin")
oddsratio(aspirin)  # odds ratio
```

```
## log odds ratios for Group and MI
##
## [1] 0.6054
```

Smoking and MI

```
library(cdabookfunc)
library(cdabookdb)
data("smoking_mi")
# oddsratio(smoking_mi, row_id = 2:1)  # odds ratio
oddsratio(smoking_mi[2:1,]) #alternatively
```

```
## log odds ratios for Smoker and MI
##
## [1] 1.341
```

2.4 Chi-Squared Tests of Independence

Example: Gender Gap in Political Party Affiliation

```
library(cdabookfunc)
library(cdabookdb)
data("gender_party")
# oddsratio(gender_party, col_id = c(1, 3))  # odds ratio
oddsratio(gender_party[,c(1,3)]) # alternatively
```

```
## log odds ratios for Gender and Party
##
## [1] 0.4729
```

Use `chisq.test()` to do chi-square test

```
# X2 test
x2_result <- chisq.test(gender_party)  # chi-square test of independence
x2_result
```

```
##
##  Pearson's Chi-squared test
##
## data:  gender_party
## X-squared = 30, df = 2, p-value = 3e-07
```

The calculation of G2 statistics needs to obtain the expected values under the assumption of independence first, and can also be obtained from the results of ‘chisq. test ()’.

```
# G2
gender_party_expected <- x2_result$expected  # obtaining the mean under the independence hypothesis
gender_party_expected
```

```
##          Party
## Gender    Democrat Independent Republican
##   Females    703.7       319.6      533.7
##   Males      542.3       246.4      411.3
```

```
Gsq <- 2 * sum(gender_party * log(gender_party / gender_party_expected))
pvalue <- 1 - pchisq(Gsq, 2)
Gsq; pvalue
```

```
## [1] 30.02
```

```
## [1] 3.034e-07
```

In addition, we can use `independent_test_of_table()` in `cdabookcode`to do X2 and G2 tests

```r
independent_test_of_table(gender_party, "X2")
```

```
## $method
## [1] "X2"
##
## $statistic
## [1] 30.07
##
## $df
## [1] 2
##
## $p.value
## [1] 2.954e-07
```

```r
independent_test_of_table(gender_party, "G2")
```

```
## $method
## [1] "G2"
##
## $statistic
## [1] 30.02
##
## $df
## [1] 2
##
## $p.value
## [1] 3.034e-07
```

Get residual and standardized residual from the result of `chisq.test()`

```r
# residual
gender_party - gender_party_expected
```

```
##          Party
## Gender    Democrat Independent Republican
##   Females   58.329       7.355    -65.683
##   Males    -58.329      -7.355     65.683
```

```r
# standardized residual
x2_result$stdres
```

```
##          Party
## Gender    Democrat Independent Republican
##   Females   4.5021      0.6995    -5.3159
##   Males    -4.5021     -0.6995     5.3159
```

2.5 Testing Independence for Ordinal Data

Example: Alcohol Use and Infant Malformation

M2 test can also use `independent_test_of_table()`

```
library(cdabookfunc)
library(cdabookdb)
data("malformation")
# compare the results of X2, G2, M2 tests
# use method="all" can do X2, G2, M2 tests at the same time
independent_test_of_table(malformation, "all", c(0, 0.5, 1.5, 4, 7), 0:1)
```

```
##      method statistic df p.value
## [1,] "X2"   12.08     4  0.01675
## [2,] "G2"   6.202     4  0.1846
## [3,] "M2"   6.57      1  0.01037
```

The choose of u and v affects result

```
independent_test_of_table(malformation, "G2", 1:5, 0:1)
```

```
## $method
## [1] "G2"
##
## $statistic
## [1] 6.202
##
## $df
## [1] 4
##
## $p.value
## [1] 0.1846
```

2.6 Exact Inference for Small Samples

Example: Fisher's Tea Tasting Colleague

```
# Calculate possibility
dhyper(0:4, 4, 4, 4)
```

```
## [1] 0.01429 0.22857 0.51429 0.22857 0.01429
```

Use `fisher.test()` to do Fisher's exact test

```
tea_tasting <- matrix(c(3, 1, 1, 3), nrow = 2)
fisher.test(tea_tasting, alternative = "g")
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  tea_tasting
## p-value = 0.2
## alternative hypothesis: true odds ratio is greater than 1
## 95 percent confidence interval:
##  0.3136    Inf
## sample estimates:
## odds ratio
##      6.408
```

```
fisher.test(tea_tasting, alternative = "t")
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  tea_tasting
## p-value = 0.5
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##    0.2117 621.9338
## sample estimates:
## odds ratio
##      6.408
```

2.7 Association in Three-Way Tables

Example: Death Penalty Verdicts and Race

```
library(cdabookfunc)
library(cdabookdb)
data("deathpenalty1")
ftable(deathpenalty1)
```

```
##                   DeathPenalty Yes  No
## Defendant Victim
## White     White                 53 414
##           Black                  0  16
## Black     White                 11  37
##          Black                 4 139
```

```
# Proportion of death sentences
prop.table(deathpenalty1, c(1, 2))[, , 1]
```

```
##          Victim
## Defendant   White   Black
##     White 0.11349 0.00000
##     Black 0.22917 0.02797
```

```
# Proportion of death sentences accoding to the race of defendant
prop.table(margin.table(deathpenalty1, margin = c(1, 3)), margin = 1)[, 1]
```

```
##   White   Black
## 0.10973 0.07853
```

```
# Odds ratio when the victim is white(conditional odds ratio)
oddsratio(deathpenalty1[, 1, ])
```

```
## log odds ratios for Defendant and DeathPenalty
##
## [1] -0.8426
```

```
# Odds ration without considering victim(marginal odds ratio)
oddsratio(margin.table(deathpenalty1, c(1, 3)))
```

```
## log odds ratios for Defendant and DeathPenalty
##
## [1] 0.3689
```

Clinical Trial

```
library(cdabookfunc)
library(cdabookdb)
data("treatment1")

# conditional odds ratio clinic=1
oddsratio(treatment1[1, ,])
```

```
## log odds ratios for Treatment and Response
##
## [1] -4.441e-16
```

```
# conditional odds ratio clinic=1
oddsratio(margin.table(treatment1, c(2, 3)))
```

```
## log odds ratios for Treatment and Response
##
```

```
## [1] 0.6931
```

Chapter 3

Generalized Linear Models

3.1 Components of a Generalized Linear Model

3.2 Generalized Linear Models for Binary Data

Example: Snoring and Heart Disease

```
library(cdabookdb)
data("snoring_heartdisease")
snoring_heartdisease
```

```
##                      Heartdisease
## Snoring                Yes   No
##   Never                 24 1355
##   Occasional            35  603
##   Nearly every night    21  192
##   Every night           30  224
```

When fitting the model to the snoring data (two-point data), you can set `family=binomial()`, where the `link` parameter of `binomial()` is `identity`, `logit`, `probit`, respectively Fitting linear probability models, logistic models, and probit models.

The following three models are fitted using the snoring frequency scores 0, 2, 4, 5, and the corresponding prediction probability is obtained.

```
scores <- c(0, 2, 4, 5)

snoring_linear <- glm(
  snoring_heartdisease ~ scores, family = binomial(link = "identity")
)
```

```r
snoring_logistics <- glm(
  snoring_heartdisease ~ scores, family = binomial(link = "logit")
)
snoring_probit <- glm(
  snoring_heartdisease ~ scores, family = binomial(link = "probit")
)

model_list <- list(snoring_linear, snoring_logistics, snoring_probit)
```

```r
# The coeffecients of the model
estimated_coef <- sapply(model_list, coef)
colnames(estimated_coef) <- c("linear", "logit", "probit")
round(estimated_coef, digits = 3)
```

```
##             linear  logit probit
## (Intercept)  0.017 -3.866 -2.061
## scores       0.020  0.397  0.188
```

The three models obtained are

$$\hat{\pi}(x) = \hat{\alpha} + \hat{\beta}x = 0.017 + 0.020x$$

$$\text{logit}(\hat{\pi}(x)) = \hat{\alpha} + \hat{\beta}x = -3.866 + 0.397x$$

$$\text{probit}(\hat{\pi}(x)) = \hat{\alpha} + \hat{\beta}x = -2.061 + 0.188x$$

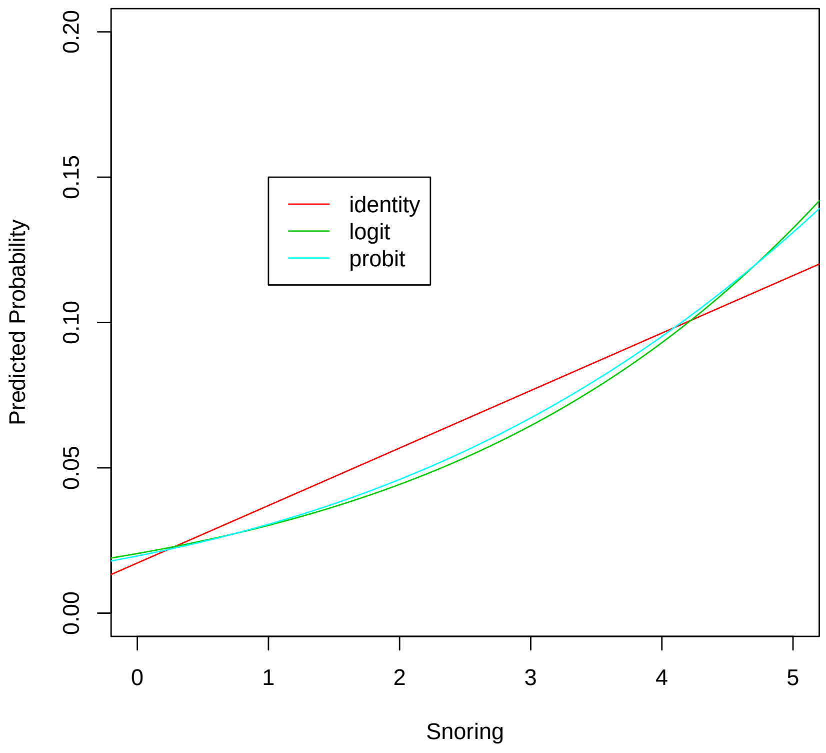
```r
# prediction of the probability
pred_prob <- sapply(model_list, predict, type = "response")
colnames(pred_prob) <- c("linear", "logit", "probit")
round(pred_prob, digits = 3)
```

```
##                    linear logit probit
## Never               0.017 0.021  0.020
## Occasional          0.057 0.044  0.046
## Nearly every night  0.096 0.093  0.095
## Every night         0.116 0.132  0.131
```

Drawing images of three models in one picture,

```r
snoring_new <- data.frame(scores=seq(-1, 6, 0.01))
plot(
  NULL,
  xlim = c(0, 5), ylim = c(0, 0.2),
  xlab = "Snoring", ylab = "Predicted Probability"
)
```

```r
line_col <- c(identity = 2, logit = 3, probit = 5)
sapply(model_list, function(m) {
    pred_result <- predict(m, snoring_new, type = "response")
    lines(
      snoring_new$scores, pred_result, type = "l",
      lty = 1, col = line_col[m$family$link]
    )
  }
)
legend(1, 0.15, names(line_col), col = line_col, lty = 1)
```



3.3 Generalized Linear Models for Count Data

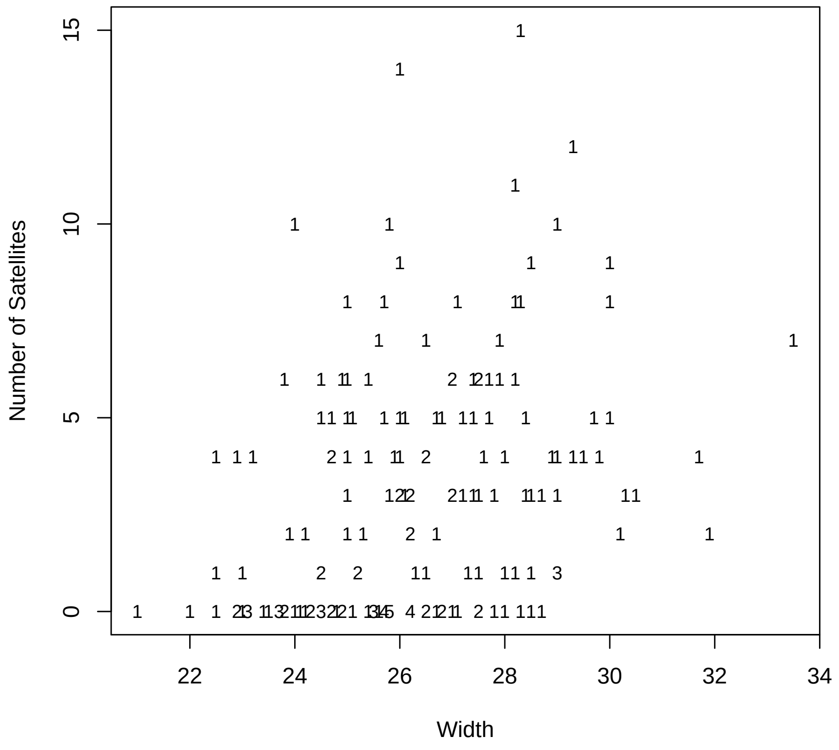
Example: Female Horseshoe Crabs and their Satellites(Poisson)

```r
library(cdabookdb)
data("horseshoecrabs") # dataset of the crabs
head(horseshoecrabs)
```

```
##   Color Spine Width Weight Satellites
## 1     2     3  28.3   3.05          8
## 2     3     3  22.5   1.55          0
## 3     1     1  26.0   2.30          9
## 4     3     3  24.8   2.10          0
## 5     3     3  26.0   2.60          4
## 6     2     3  23.8   2.10          0
```

First, an image of the response count versus width can be made, and the numbers in the figure are the number of observations of the corresponding point.

```
attach(horseshoecrabs)
tab <- table(Satellites,Width)
D <- data.frame()
for (i in 1:dim(tab)[1]) {
  for(j in 1:dim(tab)[2]){
    if(tab[i,j]){
      D=rbind(D,c(as.numeric(colnames(tab)[j]),
                  as.numeric(rownames(tab)[i]),tab[i,j]))
    }
  }
}# merge the observations with the same width and Satellites, record counts.
colnames(D) <- c('Width','Sat','counts')
plot(
  Sat ~ Width, data = D,
  pch = as.character(counts),  # set the type of points as numbers
  xlab = "Width", ylab = "Number of Satellites",  # labels of the axes
  cex = 0.8  # size of the characters
)
```



When fitting models using this data, the Poisson log-linear model can be set to `family=poisson` in `glm`, and the default association (`link`) in R-Poisson regression is logarithm, so here is not Need to modify `link`.

```
m1 <- glm(Satellites ~ Width, family = poisson(), data = horseshoecrabs)
summary(m1)
```

```
##
## Call:
## glm(formula = Satellites ~ Width, family = poisson(), data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.853  -1.988  -0.493   1.097   4.922
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -3.305      0.542   -6.09  1.1e-09 ***
## Width          0.164      0.020    8.22  < 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: 632.79  on 172  degrees of freedom
## Residual deviance: 567.88  on 171  degrees of freedom
## AIC: 927.2
##
## Number of Fisher Scoring iterations: 6
```

It can be concluded that the log-linear model of the fit is

$$\log \hat{\mu} = \hat{\alpha} + \hat{\beta}x = -3.305 + 0.164x$$

If you want to fit the Poisson model of the identity, you need to set `poisson(link="identity")`. In addition, in this case, the following error occurs when running directly back:

```
Error: no valid set of coefficients has been found: please supply starting values
In addition: Warning message:
In log(y/mu) : NaNs produced
```

That is, you need to specify the initial value of the process of finding the optimal value, otherwise you may not find the solution. Here you can use the coefficient of the log-correlated Poisson model as the initial value.

```
m2 <- glm(
  Satellites ~ Width,
  family = poisson(link = "identity"),  # Poisson model
  data = horseshoecrabs,
  start = coef(m1)  # set the coefficients of m1 as the starting values
)
summary(m2)
```

```
##
## Call:
## glm(formula = Satellites ~ Width, family = poisson(link = "identity"),
##     data = horseshoecrabs, start = coef(m1))
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.911  -1.960  -0.541   1.041   4.799
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -11.5255     0.6777   -17.0   <2e-16 ***
## Width         0.5492     0.0297    18.5   <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: 632.79  on 172  degrees of freedom
## Residual deviance: 557.71  on 171  degrees of freedom
## AIC: 917
##
## Number of Fisher Scoring iterations: 22
```

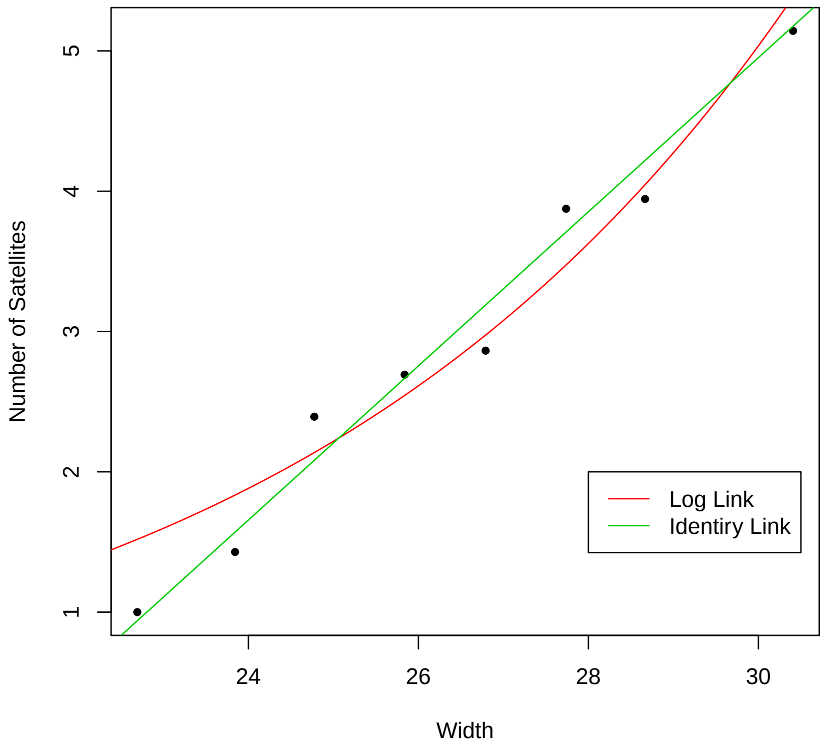
The fitted model is

$$\hat{\mu} = \hat{\alpha} + \hat{\beta}x = -11.525 + 0.549x$$

Finally, you can look at the difference in model estimates for identity and log links, Figure 3.6 in the textbook.

```r
# group by width fisrt, calculate the average width and number of Satellites
width_group <- cut(horseshoecrabs$Width,
                   breaks = c(0, 23.25 + 0:6, Inf),
                   dig.lab = 4)
mean_width_vs_sat <- sapply(levels(width_group),
                            function(x){
                              # Declare that is grouped by width_group
                              sub <- subset(horseshoecrabs,width_group==x)
                              c(mean(sub$Satellites),mean(sub$Width))
                              }
                            )
mean_satellite <- mean_width_vs_sat[1,]  # average satellite of Each group
mean_width <- mean_width_vs_sat[2,]  # average width of each group

plot(
  mean_satellite ~ mean_width,
  pch = 20, # set type of points as solid ball
  xlab = "Width", ylab = "Number of Satellites" # labels of axes
)

x <- seq(22, 32, 0.1)
y_m1 <- predict(m1, data.frame(Width = x), type = "response")
y_m2 <- predict(m2, data.frame(Width = x))
lines(x, y_m1, type = "l", col = 2)
lines(x, y_m2, type = "l", col = 3)
```

```r
legend(28, 2, c("Log Link", "Identiry Link"), col = c(2, 3), lty = 1)
```



Example: Female Horseshoe Crabs and their Satellites(Negative Binomial)

Negative binomial GLM is similar to Poisson GLM, but when setting the `family` parameter in the `glm` function, R does not have a `family` with a negative binomial distribution. You need to use `negative.binomial()` in the `MASS` package. The default `link` of this function is a logarithm, but an additional parameter `theta` is required, which means the reciprocal of `D` in Section 3.3.4 of the textbook.

The reason why you need to specify θ (should be) is that the `glm()` function does not have the process of finding the optimal θ. Here, the last reciprocal of $\widehat{D} = 1.1$ is the value for θ in the textbook.

```r
library(cdabookdb)
library(MASS)
m1 <- glm(
  Satellites ~ Width,
  family = negative.binomial(theta = 1 / 1.1),
  data = horseshoecrabs
)
summary(m1)
```

```
##
## Call:
## glm(formula = Satellites ~ Width, family = negative.binomial(theta = 1/1.1),
##     data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.782  -1.412  -0.251   0.478   2.022
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -4.0514     1.0777   -3.76  0.00023 ***
## Width         0.1920     0.0405    4.74  4.5e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(0.9091) family taken to be 0.8495)
##
##     Null deviance: 213.63  on 172  degrees of freedom
## Residual deviance: 196.33  on 171  degrees of freedom
## AIC: 755.3
##
## Number of Fisher Scoring iterations: 5
```

Another better way to fit a negative binomial GLM is to use `glm.nb()` in the `MASS` package, which has the process of finding the optimal θ without specifying θ parameter.

```r
m2 <- glm.nb(Satellites ~ Width, data = horseshoecrabs)
summary(m2)
```

```
##
## Call:
## glm.nb(formula = Satellites ~ Width, data = horseshoecrabs, init.theta = 0.90456808,
##     link = log)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.780  -1.411  -0.250   0.477   2.018
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.0525     1.1714   -3.46  0.00054 ***
## Width         0.1921     0.0441    4.36  1.3e-05 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(0.9046) family taken to be 1)
##
##     Null deviance: 213.05  on 172  degrees of freedom
## Residual deviance: 195.81  on 171  degrees of freedom
## AIC: 757.3
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  0.905
##           Std. Err.:  0.161
##
##  2 x log-likelihood:  -751.291
```

Example: British Train Accidents over Time

```r
library(cdabookdb)
library(MASS)
data("traincollisions")
head(traincollisions)
```

```
##   Year  KM Train TrRd
## 1 2003 518     0    3
## 2 2002 516     1    3
## 3 2001 508     0    4
## 4 2000 503     1    3
## 5 1999 505     1    2
## 6 1998 487     0    4
```

According to Section 3.5, when using Poisson glm to fit ratio data, the model is

$$\log(\mu/t) = \log(\mu) - \log(t) = \alpha + \beta x$$

Since the y of Poisson glm needs to be a positive integer, the log ratio ($\log(\mu/t)$) can be reduced to two logarithmic subtractions ($\log(\mu) - \log(t)$), then $\log(t)$ as `offset`.

For the `glm` function, there is a parameter `offset` that can be set directly.

```
traincollisions$year0 <- traincollisions$Year - 1975
m_poisson <- glm(
  TrRd ~ year0,
  data = traincollisions, family = poisson(),
  offset = log(traincollisions$KM)
)
summary(m_poisson)
```

```
##
## Call:
## glm(formula = TrRd ~ year0, family = poisson(), data = traincollisions,
##     offset = log(traincollisions$KM))
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.058  -0.783  -0.083   0.377   3.387
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.2114     0.1589  -26.50   <2e-16 ***
## year0        -0.0329     0.0108   -3.06   0.0022 **
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 47.376  on 28  degrees of freedom
## Residual deviance: 37.853  on 27  degrees of freedom
## AIC: 133.5
##
## Number of Fisher Scoring iterations: 5
```

The fitted model is

$$\log(\hat{\mu}) - \log(t) = -4.2114 - 0.0329x$$

The `glm.nb()` function used by the negative binary glm does not have the `offset` parameter, so it can be included in `formula` using the `offset()` function.

```
m_nb <- glm.nb(
  TrRd ~ year0 + offset(log(KM)),
  data = traincollisions
)
```

```r
summary(m_nb)
```

```
##
## Call:
## glm.nb(formula = TrRd ~ year0 + offset(log(KM)), data = traincollisions,
##     init.theta = 10.11828724, link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7237  -0.6546  -0.0587   0.3298   2.6407
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.2000     0.1958  -21.45   <2e-16 ***
## year0        -0.0337     0.0129   -2.61   0.0089 **
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(10.12) family taken to be 1)
##
##     Null deviance: 32.045  on 28  degrees of freedom
## Residual deviance: 25.264  on 27  degrees of freedom
## AIC: 132.7
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  10.12
##           Std. Err.:  8.00
##
##  2 x log-likelihood:  -126.69
```

The fitted model is

$$\log(\hat{\mu}) - \log(t) = -4.2000 - 0.0337x$$

3.4 Statistical Inference and Model Checking

Example: Snoring and Heart Disease Revisited

```r
library(cdabookdb)
data("snoring_heartdisease")
scores <- c(0, 2, 4, 5)
snoring_linear <- glm(
  snoring_heartdisease ~ scores,
  family = binomial(link = "identity")
)
summary(snoring_linear)
```

```
##
## Call:
## glm(formula = snoring_heartdisease ~ scores, family = binomial(link = "identity"))
##
## Deviance Residuals:
##              Never          Occasional  Nearly every night
##             0.0448             -0.2132              0.1101
##        Every night
##             0.0980
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.01725    0.00345    5.00  5.8e-07 ***
## scores       0.01978    0.00280    7.05  1.8e-12 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 65.904481  on 3  degrees of freedom
## Residual deviance:  0.069191  on 2  degrees of freedom
## AIC: 24.32
##
## Number of Fisher Scoring iterations: 3
```

```r
anova(snoring_linear, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: identity
##
## Response: snoring_heartdisease
##
```

```
## Terms added sequentially (first to last)
##
##
##        Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                      3       65.9
## scores  1     65.8          2        0.1  4.9e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

You can use `confint()` to get the confidence interval by the likelihood ratio method, but you need to load the `MASS` package first.

```r
library(MASS)
confint(snoring_linear)
```

```
##               2.5 %  97.5 %
## (Intercept) 0.01133 0.02483
## scores      0.01452 0.02551
```

3.5 Fitting Generalized Linear Models

Chapter 4

LOGISTIC REGRESSION

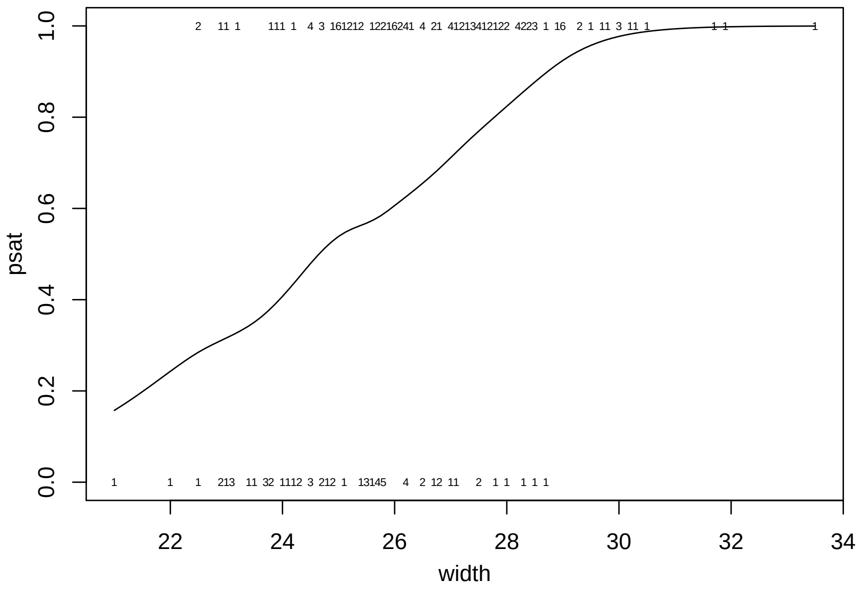
4.1 Interpreting the Logistic Regression Model

Horseshoe Crabs: Viewing and Smoothing a Binary Outcome

```
library(cdabookdb)
data("horseshoecrabs")
horseshoecrabs$psat <- as.integer(horseshoecrabs$Satellites > 0)
# generate a dummy variable psat=1 for existing at least 1 sat.
```

The following is way to draw the picture Fig. 4.2 in the textbook.

```
## figure 4.2
tab <- table(horseshoecrabs$psat,horseshoecrabs$Width)
D <- data.frame()
for (i in 1:dim(tab)[1]) {
  for(j in 1:dim(tab)[2]){
    if(tab[i,j]){
      D=rbind(D,c(as.numeric(colnames(tab)[j]),i-1,tab[i,j]))
    }
  }
}# merge the observations with the same width and past, record counts.
colnames(D) <- c('width','psat','counts')
par(mar=c(3,3,1,1.2)+0.1,mgp=c(2.2,1,0))
plot(psat~width,data=D,'n') # generate a blank canvas
text(D$width,D$psat,labels = D$counts,cex = 0.5) # add counts as text
library(gam) # general additive model
gam.fit <- gam(psat ~ s(Width), family=binomial, data=horseshoecrabs) # s = smooth funct.
curve(predict(gam.fit, data.frame(Width=x), type="resp"), add=TRUE)
```

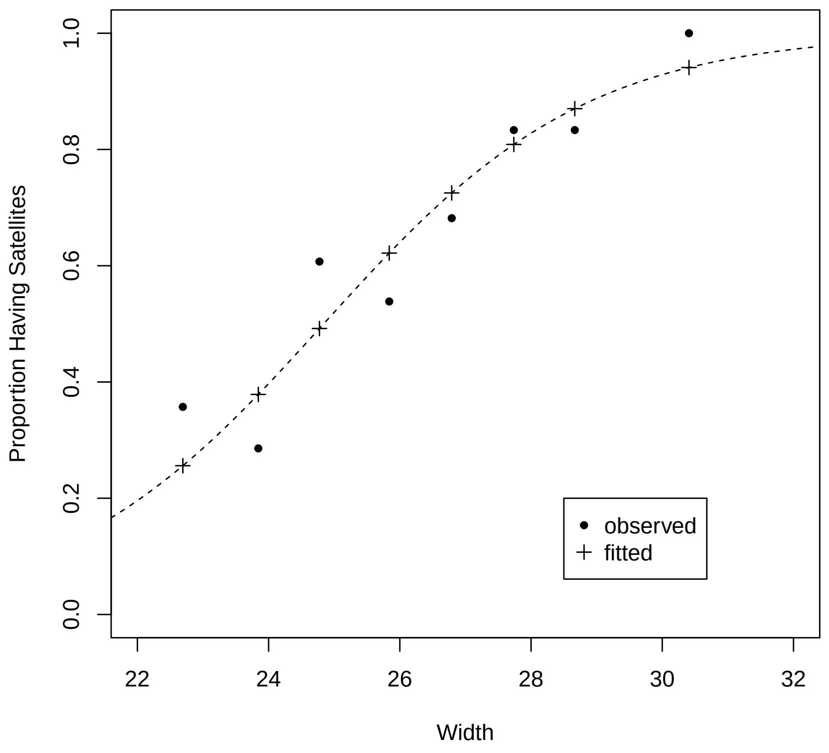


```
m1 <- glm(psat ~ Width, data = horseshoecrabs, family = binomial())
summary(m1)
```

```
##
## Call:
## glm(formula = psat ~ Width, family = binomial(), data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.028  -1.046   0.548   0.907   1.694
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -12.351      2.629   -4.70  2.6e-06 ***
## Width          0.497      0.102    4.89  1.0e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 194.45  on 171  degrees of freedom
## AIC: 198.5
##
## Number of Fisher Scoring iterations: 4
```

A detailed explanation of the model can be obtained from the textbook. The following is a mapping method for evaluating the fit of the model in the textbook (Fig. 4.3).

```
horseshoecrabs$width_group <- cut(horseshoecrabs$Width,
                                  breaks = c(0, 23.25 + 0:6, Inf),
                                  dig.lab = 4)
mean_width_vs_prop <- sapply(levels(horseshoecrabs$width_group),function(x){
  # Declare that is grouped by width_group
  sub <- subset(horseshoecrabs,width_group==x)
  c(mean(sub$psat),mean(sub$Width))
})

prop <- mean_width_vs_prop[1,]  # Propertions of Each group that has a follower
mean_width <- mean_width_vs_prop[2,]  # average width of each group

# Calculate the predicted probability under the average width of each group
pred_prop <- predict(
  m1, data.frame(Width = mean_width), type = "response"
)

# Draw the curve
width_seq <- seq(21, 33, 0.1)
pred_prop_seq <- predict(
  m1, data.frame(Width = width_seq), type = "response"
)



plot(
  prop ~ mean_width, pch = 20,  # Point type is solid dot
  xlim = c(22, 32), ylim = c(0, 1),  # Horizontal and vertical coordinate limits
  xlab = "Width", ylab = "Proportion Having Satellites"  # axes' label
)

points(mean_width, pred_prop, pch = 3)  # Point type is plus sign
points(width_seq, pred_prop_seq, type = "l", lty = 2)  # Type is line, line type is dashed
legend(28.5, 0.2, c("observed", "fitted"), pch = c(20, 3))  # legend
```



4.2 Inference for Logistic Regression

Confidence Intervals for Effects

```r
glm.fit <- glm(psat ~ Width, data = horseshoecrabs, family = binomial())
summary(glm.fit)
```

```
##
## Call:
## glm(formula = psat ~ Width, family = binomial(), data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.028  -1.046   0.548   0.907   1.694
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -12.351      2.629   -4.70  2.6e-06 ***
## Width          0.497      0.102    4.89  1.0e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 194.45  on 171  degrees of freedom
## AIC: 198.5
##
## Number of Fisher Scoring iterations: 4
```

```
glm.sum <- summary(glm.fit)$coefficients
as.vector(c(glm.sum[2,1]-1.96*glm.sum[2,2],glm.sum[2,1]+1.96*glm.sum[2,2]))
```

```
## [1] 0.2978 0.6966
```

```
# Wald confidence interval
library(car)
Anova(glm.fit) # likelihood-ratio test of width effect
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: psat
##       LR Chisq Df Pr(>Chisq)
## Width     31.3  1    2.2e-08 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
confint(glm.fit) # profile likelihood confidence interval
```

```
##                2.5 % 97.5 %
## (Intercept) -17.8100 -7.457
## Width         0.3084  0.709
```

Confidence Intervals for Probabilities

```
pred <- predict(glm.fit,
                newdata = data.frame(Width=26.5),
                type = 'resp',se.fit = T) # estimate the s.e. at the same time
as.vector(pred$fit) # \hat\pi(26.5)
```

```
## [1] 0.6955
```

```
as.vector(c(pred$fit-1.96*pred$se.fit,pred$fit+1.96*pred$se.fit)) # lower and upper bound of CI
```

```
## [1] 0.6171 0.7738
```

4.3 Logistic Regression with Categorical Predictors

Example: AZT Use and AIDS

```
library(cdabookdb)
data("AZT")
AZT0 <- as.data.frame(AZT)
# Construct dependent variable
AZT0$y <- AZT0$Symptoms == "Yes"
# fit model
AZT.glm <- glm(
  y ~ (AZTUse == "Yes") + (Race == "White"),
  data = AZT0,
  weights = Freq,
  family = binomial()
)
summary(AZT.glm)
```

```
##
## Call:
## glm(formula = y ~ (AZTUse == "Yes") + (Race == "White"), family = binomial(),
##     data = AZT0, weights = Freq)
##
## Deviance Residuals:
##     1      2      3      4      5      6      7      8
##  7.29   6.54   9.21   5.73  -5.49  -4.00  -7.07  -5.03
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -1.0736     0.2629   -4.08  4.4e-05
## AZTUse == "Yes"TRUE   -0.7195     0.2790   -2.58   0.0099
## Race == "White"TRUE    0.0555     0.2886    0.19   0.8475
##
## (Intercept)           ***
## AZTUse == "Yes"TRUE **
## Race == "White"TRUE
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 342.12  on 7  degrees of freedom
## Residual deviance: 335.15  on 5  degrees of freedom
## AIC: 341.2
##
## Number of Fisher Scoring iterations: 5
```

```
# LR test
anova(AZT.glm, test="LRT")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: y
##
## Terms added sequentially (first to last)
##
##
##                  Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                                7        342
## AZTUse == "Yes"  1     6.93         6        335   0.0085
## Race == "White"  1     0.04         5        335   0.8473
##
## NULL
## AZTUse == "Yes" **
## Race == "White"
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.4 Multiple Logistic Regression

Example: Horseshoe Crabs with Color and Width Predictors

After the data is imported from the `cdabookcode` package, since the `Color` column is a numeric type, it needs to be converted to a factor type first. In addition, when a factor-type variable is used in regression, R will use the first factor level as the baseline type. In the following example, color 4 is used as the reference type in order to be consistent with the textbook results.

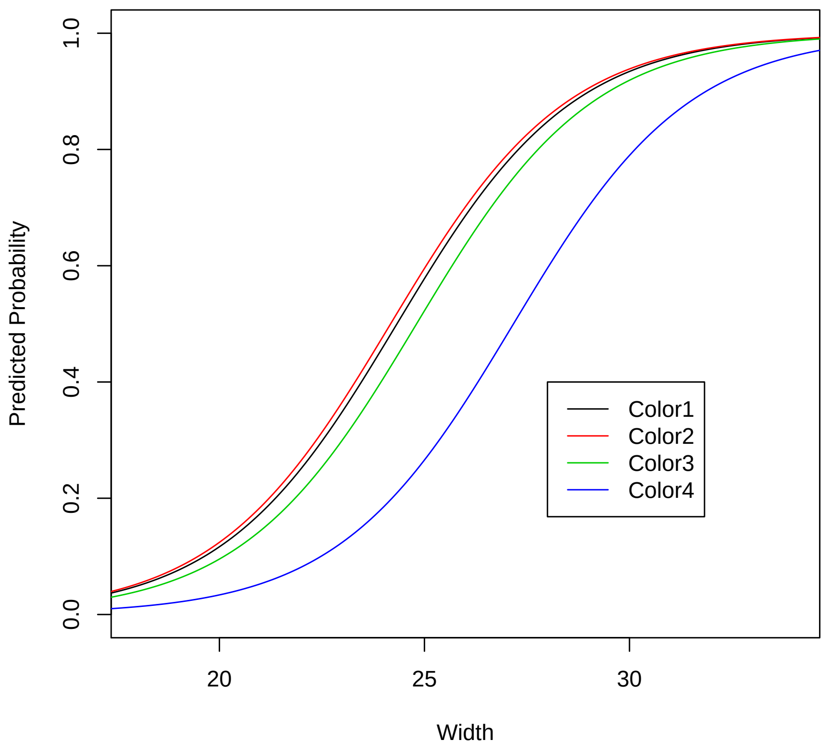
```
library(cdabookdb)
library(dplyr)
data("horseshoecrabs")
horseshoecrabs <- horseshoecrabs %>%
  mutate(
    Color_factor = factor(Color, 4:1),  # Convert Color to a factor and set the factor level
    psat = as.integer(horseshoecrabs$Satellites > 0)  # Psat for existing satellites
  )


m1 <- glm(
  psat ~ Width + Color_factor, data = horseshoecrabs, family = binomial()
)
summary(m1)
```

```
##
## Call:
## glm(formula = psat ~ Width + Color_factor, family = binomial(),
##     data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.112  -0.985   0.524   0.851   2.141
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -12.715      2.762   -4.60  4.1e-06 ***
## Width            0.468      0.106    4.43  9.3e-06 ***
## Color_factor3    1.106      0.592    1.87    0.062 .
## Color_factor2    1.402      0.548    2.56    0.011 *
## Color_factor1    1.330      0.853    1.56    0.119
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 187.46  on 168  degrees of freedom
## AIC: 197.5
##
```

```
## Number of Fisher Scoring iterations: 4
```

A detailed explanation of the model can be obtained from the textbook. The relationship between predicted probability and width in four colors is shown below (Textbook Figure 4.4)

```r
# Draw an empty plot
plot(
  NULL,  # draw no points or lines, just draw an empty plot for later use to add curves
  xlim = c(18, 34), ylim = c(0, 1),  # Horizontal and vertical coordinate limits
  xlab = "Width", ylab = "Predicted Probability"  # axes' label
)

sapply(1:4, function(i) {
  newdata <- data.frame(
    Width = seq(17, 35, 0.1),
    Color_factor = as.character(i)
  )
  pred_prop <- predict(m1, newdata, type = "response")  # Calculate the predicted probability
  points(newdata$Width, pred_prop, type = "l", col = i)  # Draw a curve
})

legend(28, 0.4, col = 1:4, legend = paste0("Color", 1:4), lty = 1)  #
```



Then consider the processing of ordered predictors in Section 4.4.3. The case in this section is similar to Section 4.4.1, but here the color variable is no longer a factor, but a score. The score here is consistent with the data set, so there is no need to do additional processing and it can be returned directly.

```
m2 <- glm(psat ~ Color + Width, family = binomial(), data = horseshoecrabs)
summary(m2)
```

```
##
## Call:
## glm(formula = psat ~ Color + Width, family = binomial(), data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.169  -0.989   0.543   0.870   1.974
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -10.071      2.807   -3.59  0.00033 ***
## Color         -0.509      0.224   -2.28  0.02286 *
## Width          0.458      0.104    4.41  1.1e-05 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 189.12  on 170  degrees of freedom
## AIC: 195.1
##
## Number of Fisher Scoring iterations: 4
```

The interaction effect is introduced in Section 4.4.4. Before fitting the model, you need to construct a dummy variable with a dark color according to the instructions in the textbook, and then fit the model containing the interaction effect.

```r
horseshoecrabs$is_dark <- as.character(horseshoecrabs$Color < 4)
# Is_dark * Width means that there are interaction term besides the two variables
# If you only want to include interaction term, you should use is_dark:Width
m3 <- glm(
  psat ~ is_dark * Width,
  family = binomial(),
  data = horseshoecrabs
)
summary(m3)
```

```
##
## Call:
## glm(formula = psat ~ is_dark * Width, family = binomial(), data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.136  -0.934   0.500   0.855   1.775
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -5.854      6.694   -0.87     0.38
## is_darkTRUE           -6.958      7.318   -0.95     0.34
## Width                  0.200      0.262    0.77     0.44
## is_darkTRUE:Width      0.322      0.286    1.13     0.26
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 186.79  on 169  degrees of freedom
## AIC: 194.8
##
## Number of Fisher Scoring iterations: 4
```

4.5 Summarizing Effects in Logistic Regression

Chapter 5

BUILDING AND APPLYING LOGISTIC REGRESSION MODELS

5.1 Strategies in Model Selection

Example: Horseshoe Crab Mating Data Revisited

In this example, the initial model has four explanatory variables: color (four categories), spine condition (three categories), weight, and width of the shell, where spine and color are factor variables. But in the `horseshoecrabs` dataset, these two are numeric variables, so we need some transformations first.

```r
library(cdabookdb)
library(dplyr)
data(horseshoecrabs)
horseshoecrabs <- horseshoecrabs %>%
  mutate(
    psat = as.integer(horseshoecrabs$Satellites > 0),
    # psat--whether to have satellites or not
    Spine_factor = factor(Spine, levels = 3:1),
    # grouping of spine, spine type 3 as the benchmark
    Color_factor = factor(Color, levels = 4:1)
    # grouping of color, color type 4 as the benchmark
  )

m1 <- glm(
  psat ~ Weight + Width + Spine_factor + Color_factor,
  family = binomial(), data = horseshoecrabs
)
summary(m1)
```

```
##
## Call:
## glm(formula = psat ~ Weight + Width + Spine_factor + Color_factor,
##     family = binomial(), data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.198  -0.942   0.485   0.849   2.120
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -9.273      3.838   -2.42   0.0157 *
## Weight           0.826      0.704    1.17   0.2407
## Width            0.263      0.195    1.35   0.1779
## Spine_factor2   -0.496      0.629   -0.79   0.4302
## Spine_factor1   -0.400      0.503   -0.80   0.4259
## Color_factor3    1.120      0.593    1.89   0.0591 .
## Color_factor2    1.506      0.567    2.66   0.0079 **
## Color_factor1    1.609      0.936    1.72   0.0855 .
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 185.20  on 165  degrees of freedom
## AIC: 201.2
##
## Number of Fisher Scoring iterations: 4
```

A likelihood-ratio test that Y is jointly independent of these predictors simultaneously tests H_0: $\beta_1 = \cdots = \beta_7 = 0$. The test statistic is $-2(L_0 - L_1) = 225.76 - 185.20 = 40.6$, $df = 172 - 165 = 7$.

```
pchisq(40.6,7,lower.tail = F)
```

```
## [1] 9.661e-07
```

So P-value=9.66e-07<0.0001.

```
attach(horseshoecrabs)
cor(Width,Weight)
```

```
## [1] 0.8869
```

```r
detach(horseshoecrabs)
```

Width and weight are highly correlated(0.887).

Example: Backward Elimination for Horseshoe Crab Data

```r
m2 <- glm(
  psat ~ Width + Spine_factor + Color_factor +
    Color_factor * Spine_factor + Width * Color_factor +
    Width * Spine_factor,
  family = binomial(),
  data = horseshoecrabs
)
summary(m2)
```

```
##
## Call:
## glm(formula = psat ~ Width + Spine_factor + Color_factor + Color_factor *
##     Spine_factor + Width * Color_factor + Width * Spine_factor,
##     family = binomial(), data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.079  -0.886   0.509   0.815   1.925
##
## Coefficients:
##                             Estimate Std. Error z value
## (Intercept)                -4.07e+00   7.27e+00   -0.56
## Width                       1.35e-01   2.83e-01    0.48
## Spine_factor2              -1.58e+01   3.96e+03    0.00
## Spine_factor1              -1.72e+01   3.96e+03    0.00
## Color_factor3              -1.63e+01   9.97e+00   -1.64
## Color_factor2              -4.05e+00   8.90e+00   -0.46
## Color_factor1              -2.01e+01   3.96e+03   -0.01
## Spine_factor2:Color_factor3  1.58e+01   3.96e+03    0.00
## Spine_factor1:Color_factor3  3.30e+01   4.48e+03    0.01
## Spine_factor2:Color_factor2  1.53e+01   3.96e+03    0.00
## Spine_factor1:Color_factor2  1.63e+01   3.96e+03    0.00
## Spine_factor2:Color_factor1  5.18e+01   6.25e+03    0.01
## Spine_factor1:Color_factor1  3.53e+01   5.59e+03    0.01
## Width:Color_factor3          6.77e-01   3.91e-01    1.73
## Width:Color_factor2          2.21e-01   3.44e-01    0.64
## Width:Color_factor1          1.21e-01   7.73e-01    0.16
## Width:Spine_factor2         -2.50e-02   6.31e-01   -0.04
## Width:Spine_factor1          8.17e-03   2.84e-01    0.03
##                             Pr(>|z|)
## (Intercept)                    0.575
## Width                          0.633
## Spine_factor2                  0.997
## Spine_factor1                  0.997
## Color_factor3                  0.102
## Color_factor2                  0.649
## Color_factor1                  0.996
## Spine_factor2:Color_factor3    0.997
## Spine_factor1:Color_factor3    0.994
## Spine_factor2:Color_factor2    0.997
## Spine_factor1:Color_factor2    0.997
## Spine_factor2:Color_factor1    0.993
## Spine_factor1:Color_factor1    0.995
## Width:Color_factor3            0.083 .
## Width:Color_factor2            0.520
## Width:Color_factor1            0.875
## Width:Spine_factor2            0.968
## Width:Spine_factor1            0.977
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 173.67  on 155  degrees of freedom
## AIC: 209.7
##
## Number of Fisher Scoring iterations: 16
```

```
# backward elimination
m2_backward <- step(m2, direction = "backward", trace = T)
```

```
## Start:  AIC=209.7
## psat ~ Width + Spine_factor + Color_factor + Color_factor * Spine_factor +
##     Width * Color_factor + Width * Spine_factor
##
##                             Df Deviance AIC
## - Spine_factor:Color_factor  6      182 206
## - Width:Spine_factor         2      174 206
## - Width:Color_factor         3      177 207
## <none>                              174 210
##
## Step:  AIC=205.6
## psat ~ Width + Spine_factor + Color_factor + Width:Color_factor +
##     Width:Spine_factor
##
##                       Df Deviance AIC
## - Width:Spine_factor  2      182 202
## - Width:Color_factor  3      186 204
## <none>                       182 206
##
## Step:  AIC=201.6
## psat ~ Width + Spine_factor + Color_factor + Width:Color_factor
##
##                       Df Deviance AIC
## - Spine_factor        2      183 199
## - Width:Color_factor  3      187 201
## <none>                       182 202
##
## Step:  AIC=199.1
## psat ~ Width + Color_factor + Width:Color_factor
##
##                       Df Deviance AIC
## - Width:Color_factor  3      188 198
## <none>                       183 199
##
## Step:  AIC=197.5
## psat ~ Width + Color_factor
##
##                 Df Deviance AIC
## <none>                 188 198
## - Color_factor  3      194 198
## - Width         1      212 220
```

```
#-C*S-S*W-S-C*W
summary(m2_backward)
```

```
##
## Call:
## glm(formula = psat ~ Width + Color_factor, family = binomial(),
##     data = horseshoecrabs)
```

```
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.112  -0.985   0.524   0.851   2.141
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -12.715      2.762   -4.60  4.1e-06 ***
## Width            0.468      0.106    4.43  9.3e-06 ***
## Color_factor3    1.106      0.592    1.87    0.062 .
## Color_factor2    1.402      0.548    2.56    0.011 *
## Color_factor1    1.330      0.853    1.56    0.119
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 187.46  on 168  degrees of freedom
## AIC: 197.5
##
## Number of Fisher Scoring iterations: 4
```

```
#C+W
```

You can try other models by yourself.

```
#C=dark+W
horseshoecrabs <- horseshoecrabs %>%
  mutate(
    color_dark=as.integer(horseshoecrabs$Color_factor != 4)
  )
m_dark <- glm(
  psat ~ Width + color_dark,
  family = binomial(),
  data = horseshoecrabs
)
summary(m_dark)
```

```
##
## Call:
## glm(formula = psat ~ Width + color_dark, family = binomial(),
##     data = horseshoecrabs)
```

```
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.082  -0.993   0.527   0.861   2.155
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -12.980      2.727   -4.76  1.9e-06 ***
## Width          0.478      0.104    4.59  4.4e-06 ***
## color_dark     1.301      0.526    2.47    0.013 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 187.96  on 170  degrees of freedom
## AIC: 194
##
## Number of Fisher Scoring iterations: 4
```

Example: Summarizing Predictive Power for Horseshoe Crab Data

Fit the model first.

```r
library(cdabookdb)
data("horseshoecrabs")
horseshoecrabs$psat <- as.integer(horseshoecrabs$Satellites > 0)
m <- glm(
  psat ~ factor(Color) + Width,
  data = horseshoecrabs, family = binomial()
)
```

Then we can obtain classification tables.

```r
pi0 <- 0.5  # cut-off value
pred_prob <- predict(m, type = "response")
pred_type <- cut(
  pred_prob, breaks = c(0, pi0, 1), labels = 0:1,
  include.lowest = TRUE
)
table(horseshoecrabs$psat, pred_type)
```

```
##    pred_type
##      0  1
##   0 31 31
##   1 15 96
```

```
attach(horseshoecrabs)
pi0 <- sum(psat)/length(psat)
detach(horseshoecrabs)
pred_prob <- predict(m, type = "response")
pred_type <- cut(
  pred_prob, breaks = c(0, pi0, 1), labels = 0:1,
  include.lowest = TRUE
)
table(horseshoecrabs$psat, pred_type)
```

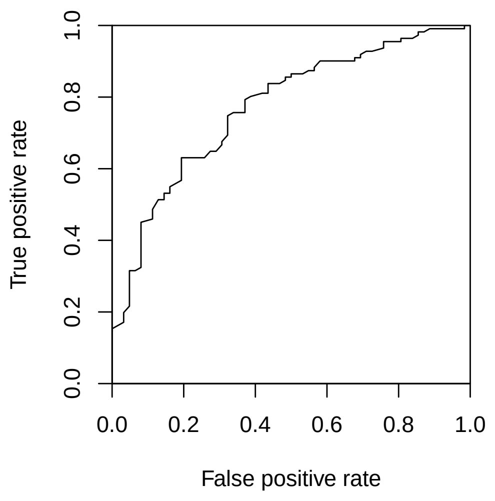
```
##    pred_type
##      0  1
##   0 43 19
##   1 36 75
```

When $\pi_0 = 0.642$, the estimated sensitivity$= 75/111 = 0.676$ and specificity$= 43/62 = 0.694$.The proportion of correct classifications is $(75 + 43)/173 = 0.682$.

Inconsistent with the textbook, with reason unidentified

We can draw the ROC curve and calculate AUC(the area under the curve) by using the function `performance` in the `ROCR` package.

```
library(ROCR)
par(pty = "s")
pred <- prediction(fitted(m), horseshoecrabs$psat)
perf <- performance(pred, "tpr", "fpr")
plot(perf, asp =1, xaxs="i", yaxs="i")
```



```r
#C+W
performance(pred,"auc")@y.values[[1]]
```

```
## [1] 0.7714
```

```r
#C=dark+W
performance(prediction(fitted(m_dark), horseshoecrabs$psat),"auc")@y.values[[1]]
```

```
## [1] 0.772
```

```r
#C
m_C <- glm(
  psat ~ factor(Color),
  data = horseshoecrabs, family = binomial()
)
performance(prediction(fitted(m_C), horseshoecrabs$psat),"auc")@y.values[[1]]
```

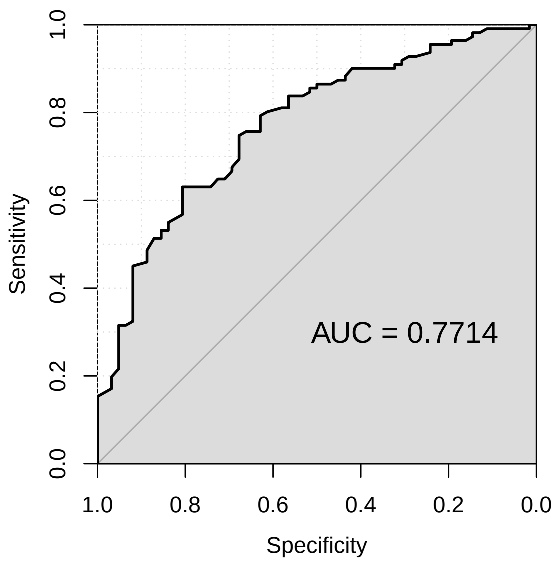
```
## [1] 0.6386
```

```r
#W
m_W <- glm(
  psat ~ Width,
  data = horseshoecrabs, family = binomial()
)
performance(prediction(fitted(m_W), horseshoecrabs$psat),"auc")@y.values[[1]]
```

```
## [1] 0.7424
```

Or by using the function `roc` in the `pROC` package. Use the command `help(plot.roc)` to see more plotting options.

```
library(pROC)
par(pty = "s")
result <- roc(
  horseshoecrabs$psat,
  predict(m, type = "response"),
  plot = TRUE,
  auc.polygon = TRUE,
  grid = TRUE,
  asp =1,
  xaxs="i",
  yaxs="i"
)
text(0.3, 0.3, labels = paste("AUC =", round(result$auc, 4)), cex = 1.3)
```



The correlation R between the observed responses $\{y_i\}$ and the model's fitted values $\{\mu_i\}$ measures predictive power.

```
#C+W
cor(horseshoecrabs$psat, fitted(m))
```

```
## [1] 0.4522
```

```
#C=dark+W
cor(horseshoecrabs$psat, fitted(m_dark))
```

```
## [1] 0.447
```

```
#C
cor(horseshoecrabs$psat, fitted(m_C))
```

```
## [1] 0.2853
```

```
#W
cor(horseshoecrabs$psat, fitted(m_W))
```

```
## [1] 0.402
```

5.2 Model Checking

Example: Likelihood-Ratio Model Comparison Tests for Horseshoe Crab Data

The following is about testing whether to include the quadratic term of width in the model.

```
library(cdabookdb)
data("horseshoecrabs")
# fit the model with and without the quadratic form respectively
m1 <- glm(
  Satellites > 0 ~ Width,
  data = horseshoecrabs, family = binomial()
)
m2 <- glm(
  Satellites > 0 ~ Width + I(Width ^ 2),
  data = horseshoecrabs, family = binomial()
)

# check the coefficient of the quadratic term
summary(m2)
```

```
##
## Call:
## glm(formula = Satellites > 0 ~ Width + I(Width^2), family = binomial(),
##     data = horseshoecrabs)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.119  -1.044   0.507   0.948   1.541
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  14.5916    30.2237    0.48     0.63
## Width        -1.5957     2.3520   -0.68     0.50
## I(Width^2)    0.0405     0.0457    0.89     0.38
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 225.76  on 172  degrees of freedom
## Residual deviance: 193.63  on 170  degrees of freedom
## AIC: 199.6
##
## Number of Fisher Scoring iterations: 5
```

```
# compare the two models (by LR test)
anova(m1, m2, test = "LR")
```

```
## Analysis of Deviance Table
##
## Model 1: Satellites > 0 ~ Width
## Model 2: Satellites > 0 ~ Width + I(Width^2)
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       171        194
## 2       170        194  1    0.825     0.36
```

Example: Goodness of Fit and the Deviance for AIDS and AZT Use Data

```
library(cdabookdb)
library(tidyr)
data("AZT")
AZT_df <- spread(as.data.frame(AZT), Symptoms, Freq)
AZT_df
```

```
##    Race AZTUse Yes No
## 1 White    Yes  14 93
## 2 White     No  32 81
## 3 Black    Yes  11 52
## 4 Black     No  12 43
```

```
m <- glm(
  cbind(Yes, No) ~ (Race == "White") + (AZTUse == "Yes"),
  data = AZT_df,
  family = binomial("logit")
)
summary(m)
```

```
##
## Call:
## glm(formula = cbind(Yes, No) ~ (Race == "White") + (AZTUse ==
##     "Yes"), family = binomial("logit"), data = AZT_df)
```

```
##
## Deviance Residuals:
##      1       2       3       4
## -0.555   0.425   0.704  -0.633
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -1.0736     0.2629   -4.08  4.4e-05
## Race == "White"TRUE   0.0555     0.2886    0.19   0.8476
## AZTUse == "Yes"TRUE  -0.7195     0.2790   -2.58   0.0099
##
## (Intercept)         ***
## Race == "White"TRUE
## AZTUse == "Yes"TRUE **
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8.3499  on 3  degrees of freedom
## Residual deviance: 1.3835  on 1  degrees of freedom
## AIC: 24.86
##
## Number of Fisher Scoring iterations: 4
```

```
m$fitted.values
```

```
##      1      2      3      4
## 0.1496 0.2654 0.1427 0.2547
```

```
# X2 and G2's df
df <- nrow(AZT_df) - length(coef(m))
df
```

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

```
# X2 test
X2 <- sum(resid(m, type = "pearson") ^ 2)
x2_pvalue <- 1- pchisq(X2, df)
c(X2 = X2, pvalue = x2_pvalue)
```

```
##     X2 pvalue
## 1.3910 0.2382
```

```r
# G2 test
G2 <- sum(resid(m, type = "deviance") ^ 2)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##     G2 pvalue
## 1.3835 0.2395
```

Example: Hosmer–Lemeshow Test for Horseshoe Crab Data

Hosmer-Lemeshow test can be realized by the function `hoslem.test()` in the `ResourceSelection` package.

```r
library(cdabookdb)
library(ResourceSelection)
data("horseshoecrabs")
horseshoecrabs$psat <- as.integer(horseshoecrabs$Satellites > 0)
m <- glm(
  psat ~ factor(Color) + Width,
  data = horseshoecrabs, family = binomial()
)
m_W <- glm(
  psat ~ Width,
  data = horseshoecrabs, family = binomial()
)
# Hosmer-Lemeshow test
# C+W
hoslem.test(m$y, fitted(m))
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  m$y, fitted(m)
## X-squared = 4.5, df = 8, p-value = 0.8
```

```r
# W only
hoslem.test(m_W$y, fitted(m_W))
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  m_W$y, fitted(m_W)
## X-squared = 4.4, df = 8, p-value = 0.8
```

Inconsistent with the textbook, maybe because the way R seperate the groups is different from SAS

Example: GraduateAdmissions at University of Florida

```r
library(cdabookdb)
library(tidyr)
data("UFAdmissions")
UFAdmissions_df <- spread(as.data.frame(UFAdmissions), Decision, Freq)
UFAdmissions_df
```

```
##    Dept Gender Admitted Rejected
## 1  anth Female       32       81
## 2  anth   Male       21       41
## 3  astr Female        6        0
## 4  astr   Male        3        8
## 5  chem Female       12       43
## 6  chem   Male       34      110
## 7  clas Female        3        1
## 8  clas   Male        4        0
## 9  comm Female       52      149
## 10 comm   Male        5       10
## 11 comp Female        8        7
## 12 comp   Male        6       12
## 13 engl Female       35      100
## 14 engl   Male       30      112
## 15 geog Female        9        1
## 16 geog   Male       11       11
## 17 geol Female        6        3
## 18 geol   Male       15        6
## 19 germ Female       17        0
## 20 germ   Male        4        1
## 21 hist Female        9        9
## 22 hist   Male       21       19
## 23 lati Female       26        7
## 24 lati   Male       25       16
## 25 ling Female       21       10
## 26 ling   Male        7        8
## 27 math Female       25       18
## 28 math   Male       31       37
## 29 phil Female        3        0
## 30 phil   Male        9        6
## 31 phys Female       10       11
## 32 phys   Male       25       53
## 33 poli Female       25       34
## 34 poli   Male       39       49
## 35 psyc Female        2      123
## 36 psyc   Male        4       41
## 37 reli Female        3        3
## 38 reli   Male        0        2
## 39 roma Female       29       13
## 40 roma   Male        6        3
## 41 soci Female       16       33
## 42 soci   Male        7       17
## 43 stat Female       23        9
## 44 stat   Male       36       14
## 45 zool Female        4       62
## 46 zool   Male       10       54
```

```
m <- glm(
  cbind(Admitted, Rejected) ~ Dept,
  data = UFAdmissions_df,
  family = binomial()
)

# X2 and G2's df
df <- nrow(UFAdmissions_df) - length(coef(m))
df
```

```
## [1] 23
```

```
# X2 test
X2 <- sum(resid(m, type = "pearson") ^ 2)
x2_pvalue <- 1- pchisq(X2, df)
c(X2 = X2, pvalue = x2_pvalue)
```

```
##       X2   pvalue
## 40.85236  0.01231
```

```
# G2 test
G2 <- sum(resid(m, type = "deviance") ^ 2)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##        G2    pvalue
## 44.735165  0.004282
```

```r
# standardized pearson residual
residuals(m,type = "pearson")/sqrt(1-hatvalues(m))
```

```
##        1        2        3        4        5        6 
## -0.76457  0.76457  2.87096 -2.87096 -0.26830  0.26830 
##        7        8        9       10       11       12 
## -1.06904  1.06904 -0.63260  0.63260  1.15752 -1.15752 
##       13       14       15       16       17       18 
##  0.94209 -0.94209  2.16641 -2.16641 -0.26082  0.26082 
##       19       20       21       22       23       24 
##  1.88730 -1.88730 -0.17627  0.17627  1.64564 -1.64564 
##       25       26       27       28       29       30 
##  1.37298 -1.37298  1.28844 -1.28844  1.34164 -1.34164 
##       31       32       33       34       35       36 
##  1.32458 -1.32458 -0.23318  0.23318 -2.27222  2.27222 
##       37       38       39       40       41       42 
##  1.26491 -1.26491  0.13970 -0.13970  0.30123 -0.30123 
##       43       44       45       46 
## -0.01229  0.01229 -1.75873  1.75873
```

```r
rstandard(m,type = "pearson")
```

```
##        1        2        3        4        5        6 
## -0.76457  0.76457  2.87096 -2.87096 -0.26830  0.26830 
##        7        8        9       10       11       12 
## -1.06904  1.06904 -0.63260  0.63260  1.15752 -1.15752 
##       13       14       15       16       17       18 
##  0.94209 -0.94209  2.16641 -2.16641 -0.26082  0.26082 
##       19       20       21       22       23       24 
##  1.88730 -1.88730 -0.17627  0.17627  1.64564 -1.64564 
##       25       26       27       28       29       30 
##  1.37298 -1.37298  1.28844 -1.28844  1.34164 -1.34164 
##       31       32       33       34       35       36 
##  1.32458 -1.32458 -0.23318  0.23318 -2.27222  2.27222 
##       37       38       39       40       41       42 
##  1.26491 -1.26491  0.13970 -0.13970  0.30123 -0.30123 
##       43       44       45       46 
## -0.01229  0.01229 -1.75873  1.75873
```

Exclude three departments “astr”, “geog” and “psyc” from the model:

```r
UFAdmissions_df_e <- UFAdmissions_df[-c(3,4,15,16,35,36),]
m_e <- glm(
  cbind(Admitted, Rejected) ~ Dept,
  data = UFAdmissions_df_e,
  family = binomial()
)

# X2 and G2's df
df <- nrow(UFAdmissions_df_e) - length(coef(m_e))
df
```

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

```
# X2 test
X2 <- sum(resid(m_e, type = "pearson") ^ 2)
x2_pvalue <- 1- pchisq(X2, df)
c(X2 = X2, pvalue = x2_pvalue)
```

```
##      X2  pvalue
## 22.7536  0.3011
```

```
# G2 test
G2 <- sum(resid(m_e, type = "deviance") ^ 2)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##      G2  pvalue
## 24.3688  0.2267
```

Add a gender effect to the model:

```
m_g <- glm(
  cbind(Admitted, Rejected) ~ Dept + Gender,
  data = UFAdmissions_df,
  family = binomial()
)

# X2 and G2's df
df <- nrow(UFAdmissions_df) - length(coef(m_g))
df
```

```
## [1] 22
```

```
# X2 test
X2 <- sum(resid(m_g, type = "pearson") ^ 2)
x2_pvalue <- 1- pchisq(X2, df)
c(X2 = X2, pvalue = x2_pvalue)
```

```
##       X2   pvalue
## 38.99080  0.01415
```

```
# G2 test
G2 <- sum(resid(m_g, type = "deviance") ^ 2)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##        G2    pvalue
## 42.360051  0.005652
```

```
exp(0.17297);exp(0.17297)-1
```

```
## [1] 1.189
```

```
## [1] 0.1888
```

This model has an ML estimate of 1.19 for the gender conditional odds ratio, the odds of admission being 19% higher for females than males, given department.

```
M <- sum(UFAdmissions_df[UFAdmissions_df[,2]=='Male',][,3])/sum(UFAdmissions_df[UFAdmissions_df[,2]=='Ma
SOR_M <- M/(1-M)
F <- sum(UFAdmissions_df[UFAdmissions_df[,2]=='Female',][,3])/sum(UFAdmissions_df[UFAdmissions_df[,2]==
SOR_F <- F/(1-F)
SOR_F/SOR_M;1-SOR_F/SOR_M
```

```
## [1] 0.9359
```

```
## [1] 0.06409
```

The marginal table collapsed over department has a sample odds ratio of 0.94, the overall odds of admission being 6% lower for females.

Example: Heart Disease and Blood Pressure

```
library(cdabookfunc)
library(cdabookdb)
data("blood_pressure")
m <- glm(
  cbind(ObservedDisease, SampleSize - ObservedDisease) ~ BloodPressure,
  data = blood_pressure,
  family = binomial()
)
summary(m)
```

```
##
## Call:
## glm(formula = cbind(ObservedDisease, SampleSize - ObservedDisease) ~
##     BloodPressure, family = binomial(), data = blood_pressure)
##
```

```
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.062  -0.598  -0.224   0.214   1.850
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -6.08203    0.72432   -8.40   <2e-16 ***
## BloodPressure  0.02434    0.00484    5.03    5e-07 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 30.0226  on 7  degrees of freedom
## Residual deviance:  5.9092  on 6  degrees of freedom
## AIC: 42.61
##
## Number of Fisher Scoring iterations: 4
```

```
round(m$fitted.values*blood_pressure$SampleSize,1)
```

```
##    1    2    3    4    5    6    7    8
##  5.2 10.6 15.1 18.1 11.6  8.9 14.2  8.4
```

About the calculation of table 5.6 in the textbook, the item `Dfbeta` is slightly different from the result obtained from using the function `dfbeta()` in R. This is because the table is calculated using SAS, and the way SAS calculates `Dfbeta` is different from R. The method SAS adopts can refer to (https://support.sas.com/documentation/cdl/en/statug/63347/HTML/default/viewer.htm#statug_logistic_sect049.htm)

Besides, there are some measures in this table unable to be calculated directly in R, such as `c` and `LR Difference`, etc. All these measures have been defined in the above documentation.

In order to calculate the above measures in the way SAS does, I define two functions `dfbetas_logit_sas()` and `influence_logit_sas()` in the package `cdabookcode`. The former uses SAS's method to calculate `Dfbeta`, and the latter calculates all the diagnostic measures in the above SAS documentation.

```
# compare `Dfbetas` between R and SAS
dfbetas_compare <- data.frame(
  R = dfbetas(m),
  SAS = dfbetas_logit_sas(m)
)
xtable::xtable(dfbetas_compare, align = "ccccc", digits = 2)
```

R..Intercept.	R.BloodPressure	SAS..Intercept.	SAS.BloodPressure
-0.61	0.56	-0.53	0.49
2.50	-2.24	1.28	-1.14
-0.41	0.34	-0.39	0.33
-0.12	0.08	-0.12	0.08
-0.00	0.01	-0.00	0.01
0.05	-0.06	0.05	-0.07
-0.33	0.38	-0.35	0.40
0.10	-0.11	0.11	-0.12

```r
# calculate all diagnostic measures
result <- influence_logit_sas(m, "data.frame")
result$`dfbetas..Intercept.` <- NULL
names(result) <- c(
  "hat", "pearson", "deviance", "dfbetas",
  "c", "cbar", "difchisq", "difdev"
)
xtable::xtable(result, align = "ccccccccc", digits = 2)
```

hat	pearson	deviance	dfbetas	c	cbar	difchisq	difdev
0.22	-0.98	-1.06	0.49	0.34	0.26	1.22	1.39
0.29	2.01	1.85	-1.14	2.26	1.62	5.64	5.04
0.26	-0.81	-0.84	0.33	0.31	0.23	0.89	0.94
0.22	-0.51	-0.52	0.08	0.09	0.07	0.33	0.34
0.13	0.12	0.12	0.01	0.00	0.00	0.02	0.02
0.13	-0.30	-0.31	-0.07	0.02	0.01	0.11	0.11
0.38	0.51	0.50	0.40	0.26	0.16	0.43	0.42
0.38	-0.14	-0.14	-0.12	0.02	0.01	0.03	0.03

```r
# standardized pearson residual
round(rstandard(m, type = "pearson"), 2)
```

```
##     1     2     3     4     5     6     7     8
## -1.11  2.37 -0.95 -0.57  0.13 -0.33  0.65 -0.18
```

```r
table <- data.frame(
  blood_pressure,
  FittedDisease = round(m$fitted.values*blood_pressure$SampleSize,1),
  StandardizedResidual = round(rstandard(m, type = "pearson"), 2)
)
xtable::xtable(table, align = "cccccc", digits = 2)
```

BloodPressure	SampleSize	ObservedDisease	FittedDisease	StandardizedResidual
111.50	156.00	3.00	5.20	-1.11
121.50	252.00	17.00	10.60	2.37
131.50	284.00	12.00	15.10	-0.95
141.50	271.00	16.00	18.10	-0.57
151.50	139.00	12.00	11.60	0.13
161.50	85.00	8.00	8.90	-0.33
176.50	99.00	16.00	14.20	0.65
191.50	43.00	8.00	8.40	-0.18

```
# X2 and G2's df
df <- nrow(blood_pressure) - length(coef(m))
df
```

```
## [1] 6
```

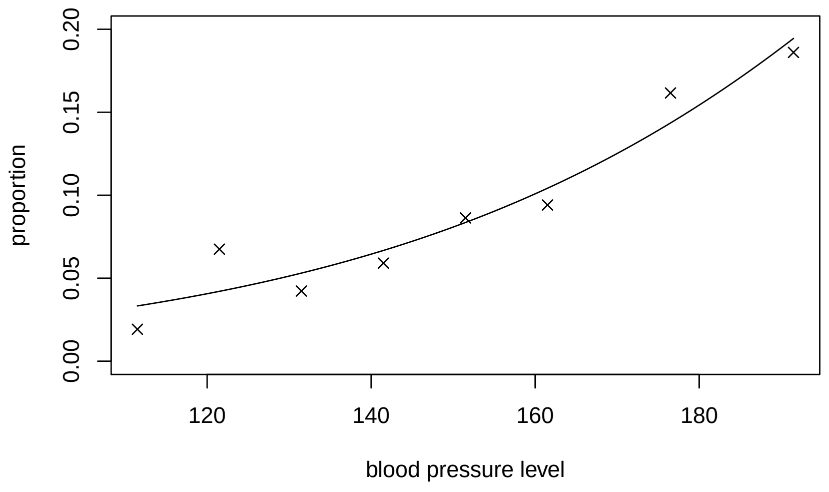
```
# X2 test
X2 <- sum(resid(m, type = "pearson") ^ 2)
x2_pvalue <- 1- pchisq(X2, df)
c(X2 = X2, pvalue = x2_pvalue)
```

```
##     X2 pvalue
## 6.2899 0.3915
```

```
# G2 test
G2 <- sum(resid(m, type = "deviance") ^ 2)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##     G2 pvalue
## 5.9092 0.4334
```

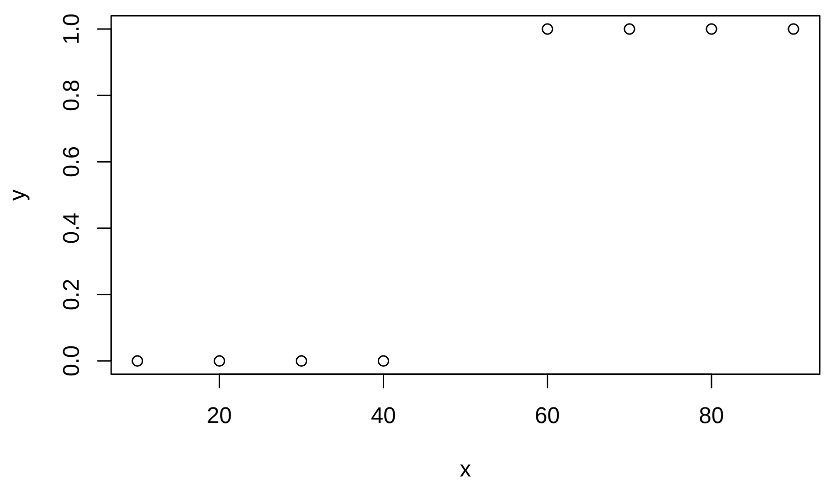
```
newdata <- data.frame(BloodPressure = seq(111.5,191.5,by=0.1))
plot(newdata$BloodPressure,predict(m,newdata = newdata,type = "response"),type = "l",xlab = "blood pres
points(blood_pressure$BloodPressure,
       blood_pressure$ObservedDisease/blood_pressure$SampleSize,
       pch = 4)
```



5.3 Effects of Sparse Data

Example: Infinite Effect Estimate: Quantitative Predictor

```
x <- c(seq(10,40,10),seq(60,90,10))
y <- c(rep(0,4),rep(1,4))
plot(x,y,xlab = "x",ylab = "y")
```



```
m <- glm(y ~ x, family = binomial("logit"))
summary(m)
```

```
##
## Call:
## glm(formula = y ~ x, family = binomial("logit"))
##
## Deviance Residuals:
##       Min         1Q     Median         3Q        Max
## -1.04e-05  -2.10e-08   0.00e+00   2.10e-08   1.04e-05
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -118.16  296046.19       0        1
## x                2.36    5805.94       0        1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1.1090e+01  on 7  degrees of freedom
## Residual deviance: 2.1827e-10  on 6  degrees of freedom
## AIC: 4
##
## Number of Fisher Scoring iterations: 25
```

Example: Clinical Trial with Sparse Data

```
library(cdabookdb)
library(tidyr)
data("treatment3")
treatment3_df1 <- as.data.frame(treatment3)
treatment3_df1$Center <- factor(treatment3_df1$Center, 5:1)
treatment3_df2 <- spread(treatment3_df1, Response, Freq)



# regress using data frame
m1_df1 <- glm(
  (Response == "Success") ~ Center + (Treatment=="Active drug"),
  family = binomial(), weights = Freq,
  data = treatment3_df1
)
summary(m1_df1)
```

```
##
## Call:
## glm(formula = (Response == "Success") ~ Center + (Treatment ==
##     "Active drug"), family = binomial(), data = treatment3_df1,
##     weights = Freq)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9488  -0.7277  -0.0001   0.5665   3.0974
##
## Coefficients:
##                                  Estimate Std. Error z value
## (Intercept)                        -2.022      0.670   -3.02
## Center4                             1.063      0.701    1.52
## Center3                           -18.614   2985.252   -0.01
## Center2                            -2.180      1.133   -1.92
## Center1                           -18.587   3180.370   -0.01
## Treatment == "Active drug"TRUE      1.546      0.702    2.20
##                                  Pr(>|z|)
## (Intercept)                        0.0025 **
## Center4                            0.1294
## Center3                            0.9950
## Center2                            0.0543 .
## Center1                            0.9953
## Treatment == "Active drug"TRUE     0.0276 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 85.77  on 14  degrees of freedom
## Residual deviance: 57.74  on  9  degrees of freedom
## AIC: 69.74
##
## Number of Fisher Scoring iterations: 17
```

```
# regress using contingency table
m1_df2 <- glm(
  cbind(Success, Failure) ~ Center + (Treatment=="Active drug"),
  family = binomial(),
  data = treatment3_df2
)
summary(m1_df2)
```

```
##
```

```
## Call:
## glm(formula = cbind(Success, Failure) ~ Center + (Treatment ==
##     "Active drug"), family = binomial(), data = treatment3_df2)
##
## Deviance Residuals:
##      1       2       3       4       5       6       7
## -0.201   0.294   0.151  -0.173   0.000   0.000   0.161
##      8       9      10
## -0.545   0.000   0.000
##
## Coefficients:
##                                Estimate Std. Error z value
## (Intercept)                      -2.022      0.670   -3.02
## Center4                           1.063      0.701    1.52
## Center3                         -22.565  21523.645    0.00
## Center2                          -2.180      1.133   -1.92
## Center1                         -22.570  23296.396    0.00
## Treatment == "Active drug"TRUE     1.546      0.702    2.20
##                               Pr(>|z|)
## (Intercept)                     0.0025 **
## Center4                         0.1294
## Center3                         0.9992
## Center2                         0.0543 .
## Center1                         0.9992
## Treatment == "Active drug"TRUE   0.0276 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 28.53202  on 9  degrees of freedom
## Residual deviance:  0.50214  on 4  degrees of freedom
## AIC: 24.86
##
## Number of Fisher Scoring iterations: 21
```

In the two models, centers 1 and 3 both have abnormally large absolute value of coefficients and SE, and coefficients are different in the two models. But the other variables are normal, and have the same coefficients and SE in the two models.

We exclude the intercept term and refit the model.

```
m2_df1 <- glm(
  (Response == "Success") ~ Center + (Treatment=="Active drug") - 1,
  family = binomial(), weights = Freq,
  data = treatment3_df1
)
summary(m2_df1)
```

```
##
## Call:
## glm(formula = (Response == "Success") ~ Center + (Treatment ==
##     "Active drug") - 1, family = binomial(), data = treatment3_df1,
##     weights = Freq)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9488  -0.7277  -0.0001   0.5665   3.0974
##
## Coefficients:
##                                Estimate Std. Error z value
## Center5                          -2.022      0.670   -3.02
## Center4                          -0.959      0.655   -1.46
## Center3                         -20.636   2985.252   -0.01
## Center2                          -4.203      1.189   -3.53
## Center1                         -20.610   3180.370   -0.01
## Treatment == "Active drug"TRUE    1.546      0.702    2.20
##                                Pr(>|z|)
## Center5                         0.00254 **
## Center4                         0.14296
## Center3                         0.99448
## Center2                         0.00041 ***
## Center1                         0.99483
## Treatment == "Active drug"TRUE  0.02757 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 130.31  on 15  degrees of freedom
## Residual deviance:  57.74  on  9  degrees of freedom
## AIC: 69.74
##
```

```
## Number of Fisher Scoring iterations: 17
```

```
m2_df2 <- glm(
  cbind(Success, Failure) ~ Center + (Treatment=="Active drug") - 1,
  family = binomial(),
  data = treatment3_df2
)
summary(m2_df2)
```

```
##
## Call:
## glm(formula = cbind(Success, Failure) ~ Center + (Treatment ==
##     "Active drug") - 1, family = binomial(), data = treatment3_df2)
##
## Deviance Residuals:
##      1       2       3       4       5       6       7
## -0.201   0.294   0.151  -0.173   0.000   0.000   0.161
##      8       9      10
## -0.545   0.000   0.000
##
## Coefficients:
##                                Estimate Std. Error z value
## Center5                          -2.022      0.670   -3.02
## Center4                          -0.959      0.655   -1.46
## Center3                         -24.587  21523.645    0.00
## Center2                          -4.203      1.189   -3.53
## Center1                         -24.592  23296.396    0.00
## Treatment == "Active drug"TRUE     1.546      0.702    2.20
##                               Pr(>|z|)
## Center5                        0.00254 **
## Center4                        0.14296
## Center3                        0.99909
## Center2                        0.00041 ***
## Center1                        0.99916
## Treatment == "Active drug"TRUE  0.02757 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 73.07369  on 10  degrees of freedom
## Residual deviance:  0.50214  on  4  degrees of freedom
## AIC: 24.86
##
## Number of Fisher Scoring iterations: 21
```

The results are similar.

Inconsistent with the textbook, possibly because of the difference in the algorithm between R and SAS

```
df <- nrow(treatment3_df2) - length(coef(m2_df2))
df
```

```
## [1] 4
```

```
G2 <- sum(resid(m2_df2, type = "deviance") ^ 2)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##     G2 pvalue
## 0.5021 0.9733
```

We delete centers 1 and 3 from the analysis.

```
treatment3_d <- treatment3_df2[-c(5,6,9,10),]
m_d <- glm(
  cbind(Success, Failure) ~ Center + (Treatment=="Active drug"),
  family = binomial(),
  data = treatment3_d
)
summary(m_d)
```

```
##
## Call:
## glm(formula = cbind(Success, Failure) ~ Center + (Treatment ==
##     "Active drug"), family = binomial(), data = treatment3_d)
##
## Deviance Residuals:
##      1       2       3       4       7       8
## -0.201   0.294   0.151  -0.173   0.161  -0.545
##
## Coefficients:
##                                  Estimate Std. Error z value
## (Intercept)                        -2.022      0.670   -3.02
## Center4                             1.063      0.701    1.52
## Center2                            -2.180      1.133   -1.92
## Treatment == "Active drug"TRUE     1.546      0.702    2.20
##                                  Pr(>|z|)
## (Intercept)                      0.0025 **
## Center4                          0.1294
## Center2                          0.0543 .
## Treatment == "Active drug"TRUE   0.0276 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 16.96288  on 5  degrees of freedom
## Residual deviance:  0.50214  on 2  degrees of freedom
## AIC: 20.86
##
## Number of Fisher Scoring iterations: 5
```

We merge centers 1, 2 and 3 and refit the model.

```
treatment3_m <- treatment3_df2[-c(7:10),]
treatment3_m[5:6,3] <- c(1,0)
treatment3_m[5:6,4] <- c(24,24)
m_m <- glm(
  cbind(Success, Failure) ~ Center + (Treatment=="Active drug"),
  family = binomial(),
  data = treatment3_m
)
summary(m_m)
```

```
##
## Call:
## glm(formula = cbind(Success, Failure) ~ Center + (Treatment ==
##     "Active drug"), family = binomial(), data = treatment3_m)
##
## Deviance Residuals:
##      1       2       3       4       5       6
## -0.208   0.305   0.143  -0.164   0.186  -0.587
##
## Coefficients:
##                                Estimate Std. Error z value
## (Intercept)                      -2.031      0.670   -3.03
## Center4                           1.065      0.702    1.52
## Center3                          -2.901      1.117   -2.60
## Treatment == "Active drug"TRUE    1.559      0.700    2.23
##                                  Pr(>|z|)
## (Intercept)                        0.0024 **
## Center4                            0.1292
## Center3                            0.0094 **
## Treatment == "Active drug"TRUE   0.0259 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27.18574  on 5  degrees of freedom
## Residual deviance:  0.56277  on 2  degrees of freedom
## AIC: 20.96
##
## Number of Fisher Scoring iterations: 5
```

Finally we try not considering the center effect.

```
treatment3_margin <- margin.table(treatment3, c(2, 3))

treatment3_margin_df <- spread(as.data.frame(treatment3_margin),Response,Freq)

m3 <- glm(
  treatment3_margin ~ (Treatment=="Active drug"),
  family = binomial(),
  data = treatment3_margin_df
)

summary(m3)
```

```
##
## Call:
## glm(formula = treatment3_margin ~ (Treatment == "Active drug"),
##     family = binomial(), data = treatment3_margin_df)
##
## Deviance Residuals:
## [1]  0  0
##
## Coefficients:
##                                  Estimate Std. Error z value
## (Intercept)                        -2.351      0.523   -4.49
## Treatment == "Active drug"TRUE    1.253      0.620    2.02
##                                Pr(>|z|)
## (Intercept)                       7e-06 ***
## Treatment == "Active drug"TRUE    0.043 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4.6054e+00  on 1  degrees of freedom
## Residual deviance: 1.0658e-14  on 0  degrees of freedom
## AIC: 11.23
##
## Number of Fisher Scoring iterations: 3
```

Now the coefficients in the model become normal.

5.4 Conditional Logistic Regression and Exact Inference

Example: Promotion Discrimination

```r
library(cdabookdb)
library(tidyr)
data("promotion_race")
promotion_race_df <- spread(as.data.frame(promotion_race), Promotion, Freq)

m <- glm(
  cbind(Yes, No) ~ Race + Month,
  data = promotion_race_df,
  family = binomial()
)

summary(m)
```

```
##
## Call:
## glm(formula = cbind(Yes, No) ~ Race + Month, family = binomial(),
##     data = promotion_race_df)
##
## Deviance Residuals:
##         1          2          3          4          5
## -9.52e-06  -1.06e-05  -7.98e-06   4.20e-08   0.00e+00
##         6
##  3.00e-08
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -25.764  52607.802    0.00     1.00
## RaceWhite         24.377  52607.802    0.00     1.00
## MonthAugust        0.208      0.800    0.26     0.80
## MonthSeptember    -0.486      0.943   -0.51     0.61
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8.2664e+00  on 5  degrees of freedom
## Residual deviance: 2.6585e-10  on 2  degrees of freedom
## AIC: 16.52
##
## Number of Fisher Scoring iterations: 23
```

The estimate for the race effect in this model turns out to be pretty extreme of -24.38.

I write a function `exact_test_for_22K`, which can do small-sample tests of conditional independence in $2 \times 2 \times K$ tables. It will return the exact p.value of the test, and note that the input should be an array.

```
library(cdabookfunc)
data <- aperm(promotion_race,c(1,3,2))
# One sided
exact_test_for_22K(data,alternative = "less")
```

```
## [1] 0.02566
```

```
# Two sided
exact_test_for_22K(data,alternative = "two.sided")
```

```
## [1] 0.05625
```

Exact conditional tests of independence for these tables can be carried out using `mantelhaen.test` in R, with argument `exact=T`.

```
mantelhaen.test(data,exact=T, alternative="less")
```

```
##
##  Exact conditional test of independence in 2 x 2 x k
##  tables
##
## data:  data
## S = 0, p-value = 0.03
## alternative hypothesis: true common odds ratio is less than 1
## 95 percent confidence interval:
##  0.0000 0.7795
## sample estimates:
## common odds ratio
##                 0
```

```
mantelhaen.test(data,exact=T, alternative="two.sided")
```

```
##
##  Exact conditional test of independence in 2 x 2 x k
##  tables
##
## data:  data
## S = 0, p-value = 0.06
## alternative hypothesis: true common odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.000 1.009
## sample estimates:
## common odds ratio
##                 0
```

Also, you can use the package `logistiX` to do the small sample test:

```
library(fastDummies)
library(logistiX)
# transfer all explanatory variables to dummy variables
promotion_race_df_1 <- dummy_cols(promotion_race_df[,-(3:4)])
promotion_race_df_2 <- cbind(promotion_race_df_1[,-(c(1:2,4,7))],promotion_race_df[,3:4])
# transfer binomial response to binary response
library(cdabookfunc)
promotion_race_df_dummy <- Binomial_To_Binary(promotion_race_df_2)
m <- logistiX(x=promotion_race_df_dummy[,1:3],y=promotion_race_df_dummy[,4])
summary(m, citype="exact", testtype="probability")
```

```
## Exact logistic regression
##
## Call:
## logistiX(x = promotion_race_df_dummy[, 1:3], y = promotion_race_df_dummy[,
##     4])
##
## Estimation method:           LX
## CI method:                   exact
## Test method:                 probability
##
```

```
## Summary of estimates, confidence intervals and parameter hypotheses tests:
##
##   estimates  2.5 %   97.5 % statistic  pvalue cardinality
## 1   -1.8813   -Inf 0.008977   0.02566 0.05625          11
## 2    0.4720 -1.646 3.015052   0.31342 0.68044           7
## 3    0.6719 -1.462 3.228892   0.27577 0.65857           7
```

We see the two-sided p-value is 0.056.

Note that the statistic is not the one-sided p-value, to see the one-sided p-value:

```
m
```

```
##    varnum method.est  estimate method.ci     lower    upper
## 1       1        MUE   -1.8813       TST -999.0000  0.00899
## 2       1        MLE -999.0000  TST-Pmid -999.0000 -0.24911
## 3       1         LX   -1.8813        SC -999.0000  0.05449
## 4       1       CCFL   -2.3275   SC-Pmid -999.0000 -0.22672
## 5       2        MUE    0.4435       TST   -1.6457  3.01506
## 6       2        MLE    0.4720  TST-Pmid   -1.4087  2.65439
## 7       2         LX    0.4720        SC   -1.3669  2.61474
## 8       2       CCFL    0.3793   SC-Pmid   -1.1505  2.24785
## 9       3        MUE    0.6438       TST   -1.4615  3.22890
## 10      3        MLE    0.6719  TST-Pmid   -1.2225  2.86692
## 11      3         LX    0.6719        SC   -1.1814  2.82788
## 12      3       CCFL    0.5798   SC-Pmid   -0.9717  2.45915
##    p-value (2-sided) p-value (LE) p-value (GE)   chi2
## 1            0.05132      0.02566       1.0000     NA
## 2            0.02566      0.01283       0.9872     NA
## 3            0.05625      0.02566       1.0000 4.5906
## 4            0.04342      0.01283       0.9872 4.5906
## 5            0.96114      0.83284       0.4806     NA
## 6            0.64773      0.67614       0.3239     NA
## 7            0.68044      0.83284       0.4806 0.2605
## 8            0.52373      0.67614       0.3239 0.2605
## 9            0.78371      0.88391       0.3919     NA
## 10           0.50794      0.74603       0.2540     NA
## 11           0.65857      0.88391       0.3919 0.5268
## 12           0.52069      0.74603       0.2540 0.5268
##          z
## 1       NA
## 2       NA
## 3  -2.1426
## 4  -2.1426
## 5       NA
## 6       NA
## 7   0.5104
## 8   0.5104
## 9       NA
## 10      NA
## 11  0.7258
## 12  0.7258
```

In the third row, We find the one-sided p-value is 0.026.

From the summary, we find the 95% CI is $(-\infty, 0.01)$, so the 95% CI of conditional odds ratio is $(e^{-\infty}, e^{0.01}) = (0, 1.01)$.

```
# pmid
summary(m, citype="pmid", testtype="probability")
```

```
## Exact logistic regression
##
## Call:
## logistiX(x = promotion_race_df_dummy[, 1:3], y = promotion_race_df_dummy[,
##     4])
##
## Estimation method:           LX
## CI method:                   pmid
## Test method:                 probability
##
## Summary of estimates, confidence intervals and parameter hypotheses tests:
##
##   estimates  2.5 %  97.5 % statistic  pvalue cardinality
## 1   -1.8813   -Inf -0.2491   0.02566 0.05625          11
## 2    0.4720 -1.409  2.6544   0.31342 0.68044           7
## 3    0.6719 -1.223  2.8669   0.27577 0.65857           7
```

The 95% CI of conditional odds ratio is $(e^{-\infty}, e^{-0.249}) = (0, 0.78)$.

Besides, the package `elrm` uses MCMC algorithm to di exact-like inference in logistic regression models:

```
library(elrm)
library(dplyr)
set.seed(5201314)
promotion_race_df_3 <- promotion_race_df_2 %>%
  mutate(
    n = Yes+No
  )
m <- elrm(formula=Yes/n~Race_Black+Month_July+Month_August, interest=~Race_Black, r=4,
    iter=40000,burnIn = 100,dataset=promotion_race_df_3);
```

```
summary(m)
```

P-value=0.057, very close to 0.056, and the 95% CI is $(-\infty, 0.016)$, also very close to the true CI.

Note: Package `logistiX` and `elrm` were both removed from the CRAN repository, you need to install them locally.

Here is the manual of `logistiX` https://cemsiis.meduniwien.ac.at/fileadmin/user_upload/_imported/fileadmin/msi_akim/CeMSIIS/KB/programme/logistiX-manual.pdf and here is a paper(with some examples on it) of `elrm` https://www.jstatsoft.org/article/view/v021i03/v21i03.pdf

The manual http://users.stat.ufl.edu/~aa/cda/Thompson_manual.pdf provides other methods(P112), such as `clogit` function in the `survival` package(we can use exact or approximate conditional likelihood), and `cond` function in the `cond` package(normal approximation), but these two methods' results are far from the textbook, so I do not give results, and you can try them by yourselves.

5.5 Sample Size and Power for Logistic Regression

Sample Size and Power for Comparing Two Proportions

To calculate the sample size required for comparing two proportions, you can use the function `samplesize_prop` in the package `cdabookcode`.

```
library(cdabookfunc)
samplesize_prop(0.2, 0.3, 0.05, 0.1)
```

```
## [1] 389
```

Sample Size Determination in Logistic Regression

To calculate sample size required in logistic regression and multiple logistic regression, you can use the function `samplesize_logit` and `samplesize_multilogit` in the package `cdabookcode`.

```
library(cdabookfunc)
samplesize_logit(0.08,0.12,0.05,0.1)
```

```
## [1] 612
```

Sample Size in Multiple Logistic Regression

```
library(cdabookfunc)
samplesize_multilogit(0.08,0.12,0.05,0.1,0.4)
```

```
## [1] 728.6
```

Chapter 6

MULTICATEGORY LOGIT MODELS

6.1 Logit Models for Nomial Responses

Example: Alligator Food Choice

```
library(VGAM)
library(cdabookdb)
data("alligators1")

c(min(alligators1$Length),max(alligators1$Length))
```

```
## [1] 1.24 3.89
```

```
# fit multicategory logit models
alligators.fit1 <- vglm(
  Food ~ Length,
  family = multinomial,
  data=alligators1
)

summary(alligators.fit1)
```

```
##
## Call:
## vglm(formula = Food ~ Length, family = multinomial, data = alligators1)
##
## Pearson residuals:
##                      Min      1Q Median     3Q   Max
## log(mu[,1]/mu[,3]) -2.33 -0.507  0.554 0.684 1.45
## log(mu[,2]/mu[,3]) -2.69 -0.482 -0.165 0.709 3.44
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept):1    1.618      1.307    1.24   0.2159
## (Intercept):2    5.697      1.794    3.18   0.0015 **
## Length:1        -0.110      0.517   -0.21   0.8314
## Length:2        -2.465      0.900      NA       NA
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,1]/mu[,3]),
## log(mu[,2]/mu[,3])
##
## Residual deviance: 98.34 on 114 degrees of freedom
##
## Log-likelihood: -49.17 on 114 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## 'Length:2'
##
##
## Reference group is level  3  of the response
```

```r
alligators.fit2 <- vglm(
  cbind(Food=="F",Food=="O",Food=="I") ~ Length,
  family = multinomial,
  data=alligators1
)

alligators.fit2
```

```
##
## Call:
## vglm(formula = cbind(Food == "F", Food == "O", Food == "I") ~
##     Length, family = multinomial, data = alligators1)
##
##
## Coefficients:
```

```
## (Intercept):1 (Intercept):2      Length:1      Length:2
##        -4.080        -5.697         2.355         2.465
##
## Degrees of Freedom: 118 Total; 114 Residual
## Residual deviance: 98.34
## Log-likelihood: -49.17
##
## This is a multinomial logit model with 3 levels
```

```
anova(alligators.fit1,type="I",test = "LRT")
```

```
## Analysis of Deviance Table (Type I tests: terms added sequentially from
## first to last)
##
## Model: 'multinomial', 'VGAMcategorical'
##
## Links: 'multilogitlink'
##
## Response: Food
##
##
##        Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                     116      115.1
## Length  2     16.8       114       98.3  0.00022 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

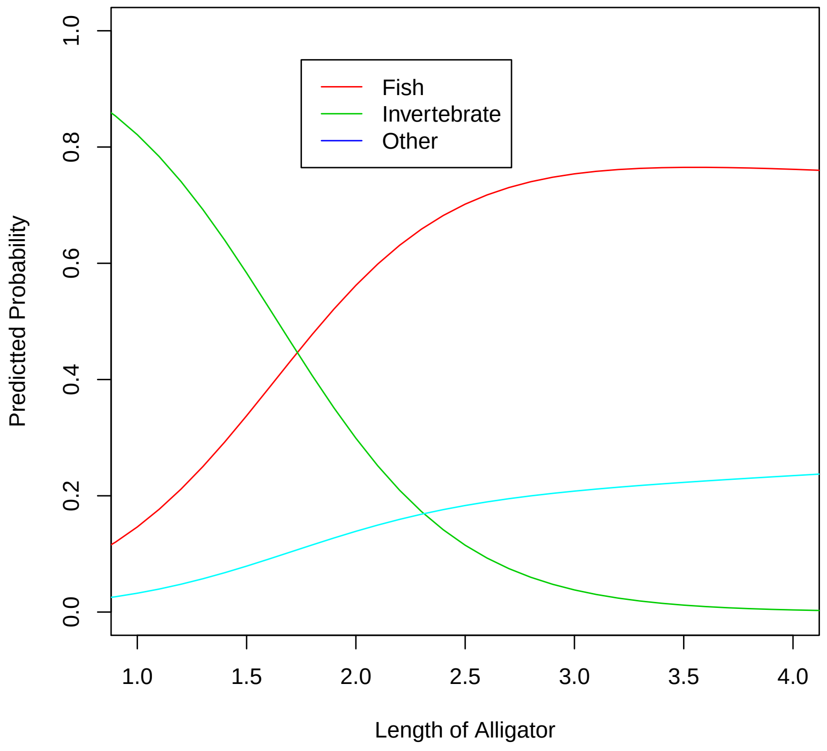
```
x <- 3.89
round(predict(alligators.fit1,data.frame(Length=x),type="response"),3)
```

```
##       F     I     O
## 1 0.763 0.005 0.232
```

Next plot the three curves of the estimated probability that the primary food type fish, invertebrate and other respectively, changing in length x.

```
new_length_x <- data.frame(Length = seq(0, 5, 0.1))
prob_food <- predict(alligators.fit1, new_length_x, type = "response")

plot(
  NULL,
  xlim = c(1, 4), ylim = c(0, 1),
  xlab = "Length of Alligator", ylab = "Predictted Probability"
)
```

```
food_col <- c(F = 2, I = 3, O = 5)

sapply(c("F", "I", "O"), function(food) {
  lines(new_length_x$Length, prob_food[, food], col = food_col[food])
})

legend(1.75, 0.95, c("Fish", "Invertebrate", "Other"), lty = 1, col = 2:5)
```



Example: Belief in Afterlife

```
library(VGAM)
library(tidyr)
library(cdabookdb)
data("afterlife2")
ftable(afterlife2)
```

```
##              Believe Yes Undecided  No
## Race  Gender
## White Female         371        49  74
##       Male           250        45  71
## Black Female          64         9  15
##       Male            25         5  13
```

```
afterlife2_df <- spread(as.data.frame(afterlife2), Believe, Freq)
afterlife2.fit1 <- vglm(
  cbind(Yes, Undecided, No) ~ (Gender == "Female") + (Race == "White"),
  data = afterlife2_df, family = multinomial()
)

summary(afterlife2.fit1)
```

```
##
## Call:
## vglm(formula = cbind(Yes, Undecided, No) ~ (Gender == "Female") +
##     (Race == "White"), family = multinomial(), data = afterlife2_df)
##
## Pearson residuals:
##   log(mu[,1]/mu[,3]) log(mu[,2]/mu[,3])
## 1             -0.219             -0.114
## 2              0.228              0.111
## 3              0.471              0.230
## 4             -0.618             -0.280
##
## Coefficients:
##                          Estimate Std. Error z value
## (Intercept):1               0.883      0.243    3.64
## (Intercept):2              -0.758      0.361   -2.10
## Gender == "Female"TRUE:1    0.419      0.171    2.44
## Gender == "Female"TRUE:2    0.105      0.247    0.43
## Race == "White"TRUE:1       0.342      0.237    1.44
## Race == "White"TRUE:2       0.271      0.354    0.77
##                          Pr(>|z|)
## (Intercept):1             0.00027 ***
## (Intercept):2             0.03593 *
## Gender == "Female"TRUE:1  0.01452 *
## Gender == "Female"TRUE:2  0.66996
## Race == "White"TRUE:1     0.14934
## Race == "White"TRUE:2     0.44416
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: log(mu[,1]/mu[,3]),
## log(mu[,2]/mu[,3])
##
## Residual deviance: 0.854 on 2 degrees of freedom
##
## Log-likelihood: -19.73 on 2 degrees of freedom
##
## Number of Fisher scoring iterations: 3
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Reference group is level  3  of the response
```

```
fitted(afterlife2.fit1)
```

```
##      Yes Undecided     No
## 1 0.7546   0.09956 0.1459
## 2 0.6783   0.12245 0.1993
## 3 0.7074   0.10018 0.1925
## 4 0.6222   0.12056 0.2573
```

```
df <- nrow(afterlife2_df)*2 - length(coef(afterlife2.fit1))
df
```

```
## [1] 2
```

```
# X2 test
X2 <- sum(resid(afterlife2.fit1, type = "pearson") ^ 2)
x2_pvalue <- 1- pchisq(X2, df)
c(X2 = X2, pvalue = x2_pvalue)
```

```
##     X2 pvalue
## 0.8609 0.6502
```

```
# G2 test
G2 <- deviance(afterlife2.fit1)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##     G2 pvalue
## 0.8539 0.6525
```

A decent fit.

```
afterlife2.fit2 <- vglm(
  cbind(Yes, Undecided, No) ~  (Race == "White"),
  data = afterlife2_df, family = multinomial()
)
anova(afterlife2.fit2,afterlife2.fit1,type = "I",test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(Yes, Undecided, No) ~ (Race == "White")
## Model 2: cbind(Yes, Undecided, No) ~ (Gender == "Female") + (Race == "White")
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         4       8.05
## 2         2       0.85  2     7.19    0.027 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
afterlife2.fit3 <- vglm(
  cbind(Yes, Undecided, No) ~  (Gender == "Female"),
  data = afterlife2_df, family = multinomial()
)
anova(afterlife2.fit3,afterlife2.fit1,type = "I",test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(Yes, Undecided, No) ~ (Gender == "Female")
## Model 2: cbind(Yes, Undecided, No) ~ (Gender == "Female") + (Race == "White")
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         4      2.848
## 2         2      0.854  2     1.99     0.37
```

```
x <- data.frame(Race="White",Gender="Female")
round(predict(afterlife2.fit1,x,type="response")[1],3)
```

```
## [1] 0.755
```

6.2 Cumulative Logit Models for Ordinal Responses

Example: Political Ideology and Party Affiliation

```
library(VGAM)
library(tidyr)
library(cdabookdb)
data("ideology")
ftable(ideology)
```

```
##                Ideology VLib SLib Mod SCon VCon
## Gender Party
## Female Dem              44   47 118   23   32
##        Rep              18   28  86   39   48
## Male   Dem              36   34  53   18   23
##        Rep              12   18  62   45   51
```

```
ide_margin <- margin.table(ideology,c(2,3))
ide_margin_df <- spread(as.data.frame(ide_margin), Ideology, Freq)
ide_m <- vglm(
  cbind(VLib, SLib, Mod, SCon, VCon) ~ Party == "Dem",
  data = ide_margin_df,
  family = cumulative(parallel = TRUE)
  # cumulative probability and the effect of x is identical for all cumulative logits
)
summary(ide_m)
```

```
##
## Call:
## vglm(formula = cbind(VLib, SLib, Mod, SCon, VCon) ~ Party ==
##     "Dem", family = cumulative(parallel = TRUE), data = ide_margin_df)
##
## Pearson residuals:
##   logitlink(P[Y<=1]) logitlink(P[Y<=2]) logitlink(P[Y<=3])
## 1              0.260             -0.170              0.808
## 2             -0.389              0.224             -0.733
##   logitlink(P[Y<=4])
## 1             -1.200
## 2              0.857
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept):1       -2.4690     0.1318  -18.73  < 2e-16 ***
## (Intercept):2       -1.4745     0.1091  -13.52  < 2e-16 ***
## (Intercept):3        0.2371     0.0948    2.50    0.012 *
## (Intercept):4        1.0695     0.1046   10.23  < 2e-16 ***
## Party == "Dem"TRUE   0.9745     0.1291    7.55  4.3e-14 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]),
## logitlink(P[Y<=2]), logitlink(P[Y<=3]), logitlink(P[Y<=4])
##
```

```
## Residual deviance: 3.688 on 3 degrees of freedom
##
## Log-likelihood: -24.62 on 3 degrees of freedom
##
## Number of Fisher scoring iterations: 3
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
## Party == "Dem"TRUE
##               2.65
```

```
pred <- predict(ide_m,data.frame(Party="Dem"),type="response")
pred
```

```
##     VLib   SLib    Mod   SCon   VCon
## 1 0.1833 0.1943 0.3931 0.1148 0.1147
```

```
pred_cum <- cumsum(pred)
pred_cum
```

```
## [1] 0.1833 0.3775 0.7706 0.8853 1.0000
```

```
anova(ide_m,type = "I",test = "LRT")
```

```
## Analysis of Deviance Table (Type I tests: terms added sequentially from
## first to last)
##
## Model: 'cumulative', 'VGAMordinal', 'VGAMcategorical'
##
## Links: 'logitlink', 'logitlink', 'logitlink', 'logitlink'
##
## Response: cbind(VLib, SLib, Mod, SCon, VCon)
##
##
##                Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                               4       62.3
## Party == "Dem"  1     58.6         3        3.7  1.9e-14
##
## NULL
## Party == "Dem" ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
df <- nrow(ide_margin_df)*4 - length(coef(ide_m))
df
```

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

```
# X2 test
X2 <- sum(resid(ide_m, type = "pearson") ^ 2)
x2_pvalue <- 1- pchisq(X2, df)
c(X2 = X2, pvalue = x2_pvalue)
```

```
##     X2 pvalue
## 3.6628 0.3002
```

```
# G2 test
G2 <- deviance(ide_m)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##     G2 pvalue
## 3.6877 0.2972
```

A decent fit.

unable to compute the score statistics

Example: Modeling Mental Health

```
library(VGAM)
library(cdabookdb)
data("impairment")

round(c(mean(impairment$LifeEvents),sd(impairment$LifeEvents)),1)
```

```
## [1] 4.3 2.7
```

```
impairment_m <- vglm(
  Impairment ~ LifeEvents + SES,
  family = cumulative(parallel = TRUE),
  # cumulative probability and the effect of x is identical for all cumulative logits
  data = impairment
)
summary(impairment_m)
```

```
##
## Call:
## vglm(formula = Impairment ~ LifeEvents + SES, family = cumulative(parallel = TRUE),
##     data = impairment)
```

```
##
## Pearson residuals:
##                        Min     1Q Median    3Q  Max
## logitlink(P[Y<=1]) -1.57 -0.705 -0.210 0.807 2.71
## logitlink(P[Y<=2]) -2.33 -0.467  0.266 0.690 1.61
## logitlink(P[Y<=3]) -3.69  0.120  0.204 0.419 1.89
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept):1   -0.282      0.623   -0.45   0.6510
## (Intercept):2    1.213      0.651    1.86   0.0625 .
## (Intercept):3    2.209      0.717    3.08   0.0021 **
## LifeEvents      -0.319      0.119   -2.67   0.0076 **
## SES              1.111      0.614    1.81   0.0704 .
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]),
## logitlink(P[Y<=2]), logitlink(P[Y<=3])
##
## Residual deviance: 99.1 on 115 degrees of freedom
##
## Log-likelihood: -49.55 on 115 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
## LifeEvents        SES
##      0.727      3.038
```

```
impairment_m_1 <- vglm(
  ordered(Impairment) ~ LifeEvents + SES + LifeEvents*SES,
  family = cumulative(parallel = TRUE),
  data = impairment
)
summary(impairment_m_1)
```

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

```
## vglm(formula = ordered(Impairment) ~ LifeEvents + SES + LifeEvents *
##     SES, family = cumulative(parallel = TRUE), data = impairment)
##
## Pearson residuals:
##                      Min     1Q Median    3Q  Max
## logitlink(P[Y<=1]) -1.39 -0.714 -0.217 0.908 2.26
## logitlink(P[Y<=2]) -2.76 -0.486  0.278 0.722 1.80
## logitlink(P[Y<=3]) -3.36  0.135  0.206 0.380 2.34
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept):1    0.0981     0.8110    0.12   0.9038
## (Intercept):2    1.5925     0.8372    1.90   0.0571 .
## (Intercept):3    2.6066     0.9097    2.87   0.0042 **
## LifeEvents      -0.4204     0.1903   -2.21   0.0272 *
## SES              0.3709     1.1302    0.33   0.7428
## LifeEvents:SES   0.1813     0.2361    0.77   0.4426
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]),
## logitlink(P[Y<=2]), logitlink(P[Y<=3])
##
## Residual deviance: 98.5 on 114 degrees of freedom
##
## Log-likelihood: -49.25 on 114 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
##     LifeEvents            SES LifeEvents:SES
##         0.6568         1.4490         1.1988
```

Std.Error of estimated coefficients inconsistent

unable to compute the score statistics

BTW, there is a manual http://users.stat.ufl.edu/~aa/cda/Thompson_manual.pdf by Dr. Laura Thompson posted on the author's website providing the use of R and S-Plus to conduct all the analyses, containing a method to compute score statistics for the proportional odds assumption using `lcr` function in the `ordinal` package (P123-124)(there isn't `lcr` function in the `ordinal` package now). But from my perspective it's more like computing LR statistics rather than score statistics, and what's more, it's result is different from that of the textbook.

```
x <- 4.3
round(predict(impairment_m,data.frame(LifeEvents=rep(x,2),SES=c(1,0)),type="response")[,1],4)
```

```
##      1      2
## 0.3678 0.1607
```

```
# lower and upper quantiles
quantile(impairment$LifeEvents, probs = c(0.25,0.75))
```

```
##  25%  75%
## 2.00 6.25
```

```
round(predict(impairment_m,data.frame(LifeEvents=c(2,2,6.5,6.5),SES=c(1,0,1,0)),type="response")[,1],4)
```

```
##      1      2      3      4
## 0.5478 0.2850 0.2239 0.0867
```

**quantile inconsistent, still use 6.5 as the upper quantile*

Invariance to Choice of Response Categories

```
library(VGAM)
library(tidyr)
library(cdabookdb)
data("ideology")
ftable(ideology)
```

```
##               Ideology VLib SLib Mod SCon VCon
## Gender Party
## Female Dem               44   47 118   23   32
##        Rep               18   28  86   39   48
## Male   Dem               36   34  53   18   23
##        Rep               12   18  62   45   51
```

```
ide_margin <- margin.table(ideology,c(2,3))
ide_margin_df <- spread(as.data.frame(ide_margin), Ideology, Freq)
ide_m <- vglm(
  cbind(VLib+SLib, Mod, SCon+VCon) ~ Party == "Dem",
  data = ide_margin_df,
  family = cumulative(parallel = TRUE)
)
summary(ide_m)
```

```
##
## Call:
## vglm(formula = cbind(VLib + SLib, Mod, SCon + VCon) ~ Party ==
##     "Dem", family = cumulative(parallel = TRUE), data = ide_margin_df)
##
## Pearson residuals:
##   logitlink(P[Y<=1]) logitlink(P[Y<=2])
## 1             -0.186              0.226
## 2              0.256             -0.182
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept):1        -1.4990     0.1108  -13.53  < 2e-16 ***
## (Intercept):2         0.2135     0.0961    2.22    0.026 *
## Party == "Dem"TRUE    1.0059     0.1322    7.61  2.7e-14 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y<=1]),
## logitlink(P[Y<=2])
##
## Residual deviance: 0.185 on 1 degrees of freedom
##
## Log-likelihood: -12.4 on 1 degrees of freedom
##
## Number of Fisher scoring iterations: 3
##
## No Hauck-Donner effect found in any of the estimates
##
##
## Exponentiated coefficients:
## Party == "Dem"TRUE
##              2.734
```

6.3 Paired-Category Ordinal Logits

Example: Political Ideology Revisited

```r
library(VGAM)
library(tidyr)
library(cdabookdb)
data("ideology")
ftable(ideology)
```

```
##               Ideology VLib SLib Mod SCon VCon
## Gender Party
## Female Dem               44   47 118   23   32
##        Rep               18   28  86   39   48
## Male   Dem               36   34  53   18   23
##        Rep               12   18  62   45   51
```

```r
ideology_df <- spread(as.data.frame(ideology), Ideology, Freq)
ide_margin <- margin.table(ideology,c(2,3))
ide_margin_df <- spread(as.data.frame(ide_margin), Ideology, Freq)

ide_m <- vglm(
  cbind(VLib, SLib, Mod, SCon, VCon) ~ Party == "Dem",
  data = ide_margin_df,
  # Adjacent-Categories Logits
  # The effects of x on the odds of making the higher instead of the lower response
  # are identical for each pair of adjacent response categories
  family = acat(reverse = TRUE, parallel = TRUE)
)
summary(ide_m)
```

```
##
## Call:
## vglm(formula = cbind(VLib, SLib, Mod, SCon, VCon) ~ Party ==
##     "Dem", family = acat(reverse = TRUE, parallel = TRUE), data = ide_margin_df)
##
## Pearson residuals:
##   loglink(P[Y=1]/P[Y=2]) loglink(P[Y=2]/P[Y=3])
## 1                -0.0253                 0.0541
## 2                 0.0196                -0.0814
##   loglink(P[Y=3]/P[Y=4]) loglink(P[Y=4]/P[Y=5])
## 1                  1.034                  -1.48
## 2                 -0.917                   1.15
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept):1         -0.439      0.140   -3.14   0.0017 **
## (Intercept):2         -1.172      0.112  -10.46  < 2e-16 ***
## (Intercept):3          0.732      0.109    6.72  1.8e-11 ***
## (Intercept):4        -0.368      0.121   -3.03   0.0025 **
## Party == "Dem"TRUE    0.435      0.060    7.25  4.1e-13 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: loglink(P[Y=1]/P[Y=2]),
## loglink(P[Y=2]/P[Y=3]), loglink(P[Y=3]/P[Y=4]),
## loglink(P[Y=4]/P[Y=5])
##
## Residual deviance: 5.524 on 3 degrees of freedom
##
## Log-likelihood: -25.54 on 3 degrees of freedom
##
## Number of Fisher scoring iterations: 4
##
## No Hauck-Donner effect found in any of the estimates
```

```r
df <- nrow(ide_margin_df)*4 - length(coef(ide_m))
df
```

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

```r
# G2 test
G2 <- deviance(ide_m)
g2_pvalue <- 1 - pchisq(G2, df)
c(G2 = G2, pvalue = g2_pvalue)
```

```
##     G2 pvalue
## 5.5238 0.1372
```

It's a decent fit.

```r
ide_m_1 <- vglm(
  cbind(VLib, SLib, Mod, SCon, VCon) ~ 1,
  data = ide_margin_df,
  family = acat(reverse = TRUE, parallel = TRUE)
)
anova(ide_m_1,ide_m,type = "I",test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(VLib, SLib, Mod, SCon, VCon) ~ 1
## Model 2: cbind(VLib, SLib, Mod, SCon, VCon) ~ Party == "Dem"
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1         4       62.3
## 2          3         5.5  1     56.8  4.8e-14 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Example: A Developmental Toxicity Study

```
library(VGAM)
library(tidyr)
library(cdabookdb)
data("toxicity")

toxicity_df <- spread(as.data.frame(toxicity),Response,Freq)
concentration <- as.numeric(rownames(toxicity))
toxicity_df$Concentration <- concentration
m1 <- glm(
  cbind(`Non-live`,Malformation+Normal) ~ concentration,
  family=binomial("logit"),
  data = toxicity_df
)
summary(m1)
```

```
##
## Call:
## glm(formula = cbind(`Non-live`, Malformation + Normal) ~ concentration,
##     family = binomial("logit"), data = toxicity_df)
##
## Deviance Residuals:
##      1       2       3       4       5
##  1.132   1.017  -0.597  -1.646   0.628
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -3.247934   0.157660   -20.6   <2e-16 ***
## concentration  0.006389   0.000435    14.7   <2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 259.1073  on 4  degrees of freedom
## Residual deviance:   5.7775  on 3  degrees of freedom
## AIC: 35.2
##
## Number of Fisher Scoring iterations: 4
```

```
m2 <- glm(
  cbind(Malformation,Normal) ~ concentration,
  family=binomial("logit"),
  data = toxicity_df
)
summary(m2)
```

```
##
## Call:
## glm(formula = cbind(Malformation, Normal) ~ concentration, family = binomial("logit"),
##     data = toxicity_df)
##
## Deviance Residuals:
##       1        2        3        4        5
##  0.0628  -2.1047  -0.4551   0.8515  -0.8337
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -5.70190    0.33225   -17.2   <2e-16 ***
## concentration  0.01737    0.00123    14.2   <2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 652.5831  on 4  degrees of freedom
## Residual deviance:   6.0609  on 3  degrees of freedom
## AIC: 25.49
##
## Number of Fisher Scoring iterations: 4
```

```
m3 <- vglm(
  cbind(`Non-live`,Malformation,Normal) ~ concentration,
  family=cratio(reverse = FALSE, parallel = FALSE),
  data = toxicity_df
)
summary(m3)
```

```
##
## Call:
## vglm(formula = cbind(`Non-live`, Malformation, Normal) ~ concentration,
##     family = cratio(reverse = FALSE, parallel = FALSE), data = toxicity_df)
##
## Pearson residuals:
##   logitlink(P[Y>1|Y>=1]) logitlink(P[Y>2|Y>=2])
## 1                 -1.190                 -0.063
## 2                 -1.060                  1.480
## 3                  0.586                  0.446
## 4                  1.596                 -0.879
## 5                 -0.629                  0.858
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept):1    3.247934   0.157660    20.6   <2e-16 ***
## (Intercept):2    5.701902   0.330652    17.2   <2e-16 ***
## concentration:1 -0.006389   0.000435   -14.7   <2e-16 ***
## concentration:2 -0.017375   0.001213   -14.3   <2e-16 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Names of linear predictors: logitlink(P[Y>1|Y>=1]),
## logitlink(P[Y>2|Y>=2])
##
## Residual deviance: 11.84 on 6 degrees of freedom
##
## Log-likelihood: -26.35 on 6 degrees of freedom
##
## Number of Fisher scoring iterations: 5
##
## Warning: Hauck-Donner effect detected in the following estimate(s):
## '(Intercept):1', 'concentration:2'
```

Here the sign is opposite to that in the textbook: what the textbook compute is $\text{logit}(P[Y = 1|Y >= 1])$ and $\text{logit}(P[Y = 2|Y >= 2])$, and here R compute $\text{logit}(P[Y > 1|Y >= 1])$ and $\text{logit}(P[Y > 2|Y >= 2])$.

```r
# G2 for m1
G2_1 <- deviance(m1)
G2_1
```

```
## [1] 5.777
```

```
df_1 <- nrow(toxicity_df) - length(coef(m1))
df_1
```

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

```
# G2 for m2
G2_2 <- deviance(m2)
G2_2
```

```
## [1] 6.061
```

```
df_2 <- nrow(toxicity_df) - length(coef(m2))
df_2
```

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

```
# G2 for m3
G2_3 <- deviance(m3)
G2_3
```

```
## [1] 11.84
```

```
G2_1+G2_2
```

```
## [1] 11.84
```

```
df_3 <- nrow(toxicity_df)*2 - length(coef(m3))
df_3
```

```
## [1] 6
```

```
df_1+df_2
```

```
## [1] 6
```

```
g2_pvalue <- 1 - pchisq(G2_3, df_3)
c(G2 = G2_3, pvalue = g2_pvalue)
```

```
##       G2   pvalue
## 11.83839  0.06567
```

6.4 Tests of Conditional Independence

Example: Job Satisfaction and Income

First consider cumulative logit models:

```
library(VGAM)
library(cdabookdb)
data("job_satisfaction2")
```

```r
library(tidyr)

job_df <- spread(as.data.frame(job_satisfaction2),JobSatisfaction,Freq)

job_df$Income <- rep(c(3,10,20,35),2)

# the model with an income effect
m1 <- vglm(
  cbind(`Very Dissatisfied`,`A Little Satisfied`,`Moderately Satisfied`,`Very Satisfied`) ~ Income + Ge
  family=cumulative(parallel = TRUE),
  data = job_df
)

# without
m2 <- vglm(
  cbind(`Very Dissatisfied`,`A Little Satisfied`,`Moderately Satisfied`,`Very Satisfied`) ~ Gender,
  family=cumulative(parallel = TRUE),
  data = job_df
)

anova(m2,m1,type = "I", test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(`Very Dissatisfied`, `A Little Satisfied`, `Moderately Satisfied`,
##     `Very Satisfied`) ~ Gender
## Model 2: cbind(`Very Dissatisfied`, `A Little Satisfied`, `Moderately Satisfied`,
##     `Very Satisfied`) ~ Income + Gender
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1        20       19.6
## 2        19       13.9  1     5.67    0.017 *
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Next consider baseline-category logits models, and also treat income as nominal :

```r
# the model with an income effect
m3 <- vglm(
  cbind(`Very Dissatisfied`,`A Little Satisfied`,`Moderately Satisfied`,`Very Satisfied`)
  ~ Gender + factor(Income),
  family = multinomial(),
  data = job_df
)

# without
m4 <- vglm(
  cbind(`Very Dissatisfied`,`A Little Satisfied`,`Moderately Satisfied`,`Very Satisfied`)
  ~ Gender,
  family = multinomial(),
  data = job_df
)



anova(m4,m3,type = "I",test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: cbind(`Very Dissatisfied`, `A Little Satisfied`, `Moderately Satisfied`,
##     `Very Satisfied`) ~ Gender
## Model 2: cbind(`Very Dissatisfied`, `A Little Satisfied`, `Moderately Satisfied`,
##     `Very Satisfied`) ~ Gender + factor(Income)
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1        18      19.37
## 2         9       7.09  9     12.3      0.2
```

Generalized Cochran–Mantel–Haenszel Tests

```r
library(vcdExtra)
library(cdabookfunc)
x <- aperm(job_satisfaction2,c(2,3,1))

# sample correlation between income and job satisfaction for females
r_compute(x[,,1],u=c(3,10,20,35),v=c(1,3,4,5))
```

```
## [1] 0.1601
```

```r
# for males
r_compute(x[,,2],u=c(3,10,20,35),v=c(1,3,4,5))
```

```
## [1] 0.371
```

```r
# genaralized CMH test
CMHtest(x, rscores = c(3,10,20,35), cscores = c(1,3,4,5),overall = TRUE)
```

```
## $`Gender:Female`
## Cochran-Mantel-Haenszel Statistics for Income by JobSatisfaction
##  in stratum Gender:Female
##
##                  AltHypothesis Chisq Df  Prob
## cor        Nonzero correlation  1.62  1 0.204
## rmeans  Row mean scores differ  3.93  3 0.269
## cmeans  Col mean scores differ  2.96  3 0.398
## general    General association  6.71  9 0.667
##
##
## $`Gender:Male`
## Cochran-Mantel-Haenszel Statistics for Income by JobSatisfaction
##  in stratum Gender:Male
##
##                  AltHypothesis Chisq Df   Prob
## cor        Nonzero correlation  5.37  1 0.0205
## rmeans  Row mean scores differ  7.06  3 0.0702
## cmeans  Col mean scores differ  5.76  3 0.1240
## general    General association 13.88  9 0.1267
##
##
## $ALL
## Cochran-Mantel-Haenszel Statistics for Income by JobSatisfaction
##  Overall tests, controlling for all strata
##
##                  AltHypothesis Chisq Df   Prob
## cor        Nonzero correlation  6.16  1 0.0131
## rmeans  Row mean scores differ  9.03  3 0.0288
## cmeans  Col mean scores differ  6.38  3 0.0946
## general    General association  10.2  9  0.335
```

The generalized correlation statistics is 6.1563, and df = 1(P = 0.013094).

```
# sample correlation between income and job satisfaction for females
r_compute(x[,,1],u=1:4,v=1:4)
```

```
## [1] 0.1709
```

```
# for males
r_compute(x[,,2],u=1:4,v=1:4)
```

```
## [1] 0.3814
```

```
# genaralized CMH test
CMHtest(x, rscores = 1:4, cscores = 1:4,overall = TRUE)
```

```
## $`Gender:Female`
## Cochran-Mantel-Haenszel Statistics for Income by JobSatisfaction
##  in stratum Gender:Female
##
##                  AltHypothesis Chisq Df  Prob
## cor        Nonzero correlation  1.84  1 0.175
## rmeans  Row mean scores differ  4.16  3 0.244
## cmeans  Col mean scores differ  2.97  3 0.396
## general    General association  6.71  9 0.667
##
##
## $`Gender:Male`
## Cochran-Mantel-Haenszel Statistics for Income by JobSatisfaction
##  in stratum Gender:Male
##
##                  AltHypothesis Chisq Df   Prob
## cor        Nonzero correlation  5.67  1 0.0172
## rmeans  Row mean scores differ  6.58  3 0.0866
## cmeans  Col mean scores differ  6.79  3 0.0790
## general    General association 13.88  9 0.1267
##
##
## $ALL
## Cochran-Mantel-Haenszel Statistics for Income by JobSatisfaction
##  Overall tests, controlling for all strata
##
##                  AltHypothesis Chisq Df   Prob
## cor        Nonzero correlation  6.62  1 0.0101
## rmeans  Row mean scores differ  9.23  3 0.0264
## cmeans  Col mean scores differ  6.96  3 0.0732
## general    General association  10.2  9  0.335
```

The generalized correlation statistics is 6.6235, and df = 1(P = 0.010064).

BTW, here is a mistake on the generalized correlation statistics in the Chinese edition. It writes 6.0 but it is actually 6.6.

```r
scores <- 1:4
round(apply(x, 3, function(m){
  m%*%scores
})/apply(x,3,rowSums),2)
```

```
##       Gender
##        Female Male
##   [1,]   2.82 2.60
##   [2,]   2.84 2.78
##   [3,]   3.29 3.30
##   [4,]   3.00 3.31
```

The generalized CMH statistic for testing whether the true row mean scores differ equals 9.2259 with df = 3(P = 0.026434).

The general association statistic equals 10.2, with df = 9(P = 0.33453)

Chapter 7

LOGLINEAR MODELS FOR CONTINGENCY TABLES

```r
library(cdabookdb)
library(cdabookfunc)
library(MASS)
```

7.1 Loglinear Models for Two-Way and Three-Way Tables

Cross-Classification of Race by Belief in Life after Death

```r
data(afterlife3)
```

There are three ways to fit a log-linear model: the `glm` function and the `loglin` function in the `stats` package, and the `loglm` function in the `MASS` package.

Independent log-linear model fitting:

The `glm` function can be set so that the parameter of the last category is equal to 0. The likelihood ratio test and the Pearson test are required to use the external function. See 2.4 for more.

```r
independent_test_of_table(afterlife3, "G2")
```

```
## $method
## [1] "G2"
##
## $statistic
## [1] 0.3565
##
## $df
## [1] 2
```

```
##
## $p.value
## [1] 0.8367
```

```
independent_test_of_table(afterlife3, "X2")
```

```
## $method
## [1] "X2"
##
## $statistic
## [1] 0.3601
##
## $df
## [1] 2
##
## $p.value
## [1] 0.8352
```

```
afterlife3<-as.data.frame(afterlife3)
afterlife3$Race <- relevel(afterlife3$Race,ref = "Other") #set the defualt
afterlife3$Belief <- relevel(afterlife3$Belief,ref = "No or Undecided")
life_glm1<-glm(afterlife3$Freq ~ afterlife3$Race+afterlife3$Belief, family=poisson())
summary(life_glm1)
```

```
##
## Call:
## glm(formula = afterlife3$Freq ~ afterlife3$Race + afterlife3$Belief,
##     family = poisson())
##
## Deviance Residuals:
##       1        2        3        4        5        6
## -0.0172   0.1578  -0.2019   0.0363  -0.3369   0.4192
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)            3.0003     0.1061    28.3   <2e-16
## afterlife3$RaceWhite   2.7014     0.0985    27.4   <2e-16
## afterlife3$RaceBlack   1.0521     0.1107     9.5   <2e-16
## afterlife3$BeliefYes   1.4985     0.0570    26.3   <2e-16
##
## (Intercept)          ***
## afterlife3$RaceWhite ***
## afterlife3$RaceBlack ***
## afterlife3$BeliefYes ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 2849.21758  on 5  degrees of freedom
## Residual deviance:    0.35649  on 2  degrees of freedom
## AIC: 49.44
##
## Number of Fisher Scoring iterations: 3
```

The `loglin` function uses a method in which the parameter sum of each factor is 0, so the fitted model coefficients are different from the results in the book, but the odds ratio and the fitted value are not affected, and `loglin` can directly get X^2 And the value of G^2.

```
data(afterlife3)
life_loglin1<-loglin(afterlife3,margin=list(1,2),param=TRUE)
```

```
## 2 iterations: deviation 2.274e-13
```

```
life_loglin1$lrt #G^2
```

```
## [1] 0.3565
```

```
life_loglin1$pearson #X^2
```

```
## [1] 0.3601
```

```
life_loglin1$df  #df
```

```
## [1] 2
```

```
life_loglin1$param  #coefficients
```

```
## $`(Intercept)`
## [1] 5.001
##
## $Race
##   White   Black   Other
##  1.4502 -0.1991 -1.2512
##
## $Belief
##             Yes No or Undecided
##          0.7492         -0.7492
```

`Loglm` is the same as `loglin`. The method used is that the summation of the parameter of each factor is 0. The fitted model coefficients are different from the results on the book. At the same time, the results of the Pearson test and the likelihood ratio test can be directly obtained.

```
life_loglm1<-loglm(~Race+Belief,afterlife3,fitted=TRUE)  #or (accident_loglm1<-loglm(~1+2,afterlife3))
life_loglm1
```

```
## Call:
## loglm(formula = ~Race + Belief, data = afterlife3, fitted = TRUE)
##
## Statistics:
##                      X^2 df P(> X^2)
## Likelihood Ratio 0.3565  2   0.8367
## Pearson          0.3601  2   0.8352
```

```
life_loglm1$param                  #coefficients
```

```
## $`(Intercept)`
## [1] 5.001
##
## $Race
##   White   Black   Other
##  1.4502 -0.1991 -1.2512
##
## $Belief
##             Yes No or Undecided
##          0.7492         -0.7492
```

Saturated log-linear model fit:

```
afterlife3<-as.data.frame(afterlife3)
afterlife3$Race <- relevel(afterlife3$Race,ref = "Other") #set the defualt
afterlife3$Belief <- relevel(afterlife3$Belief,ref = "No or Undecided")
life_glm2<-glm(afterlife3$Freq ~ afterlife3$Race+afterlife3$Belief+afterlife3$Race*afterlife3$Belief, f
summary(life_glm2)
```

```
##
## Call:
## glm(formula = afterlife3$Freq ~ afterlife3$Race + afterlife3$Belief +
##     afterlife3$Race * afterlife3$Belief, family = poisson())
##
## Deviance Residuals:
## [1]  0  0  0  0  0  0
##
## Coefficients:
##                                            Estimate
## (Intercept)                                   3.091
## afterlife3$RaceWhite                         2.613
## afterlife3$RaceBlack                         0.916
## afterlife3$BeliefYes                         1.386
## afterlife3$RaceWhite:afterlife3$BeliefYes    0.110
## afterlife3$RaceBlack:afterlife3$BeliefYes    0.167
##                                           Std. Error
## (Intercept)                                   0.213
## afterlife3$RaceWhite                          0.221
## afterlife3$RaceBlack                          0.252
## afterlife3$BeliefYes                          0.238
## afterlife3$RaceWhite:afterlife3$BeliefYes     0.247
## afterlife3$RaceBlack:afterlife3$BeliefYes     0.281
##                                           z value Pr(>|z|)
## (Intercept)                                 14.50  < 2e-16
## afterlife3$RaceWhite                        11.83  < 2e-16
## afterlife3$RaceBlack                         3.63  0.00028
## afterlife3$BeliefYes                         5.82    6e-09
## afterlife3$RaceWhite:afterlife3$BeliefYes    0.44  0.65695
## afterlife3$RaceBlack:afterlife3$BeliefYes    0.59  0.55189
##
## (Intercept)                               ***
## afterlife3$RaceWhite                      ***
## afterlife3$RaceBlack                      ***
## afterlife3$BeliefYes                      ***
## afterlife3$RaceWhite:afterlife3$BeliefYes
## afterlife3$RaceBlack:afterlife3$BeliefYes
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance:  2.8492e+03  on 5  degrees of freedom
## Residual deviance: -9.4813e-14  on 0  degrees of freedom
## AIC: 53.08
##
## Number of Fisher Scoring iterations: 3
```

```
data(afterlife3)
life_loglin2<-loglin(afterlife3,margin=list(c(1,2)),param=TRUE)
```

```
## 2 iterations: deviation 0
```

```
life_loglin2$param  #coefficients
```

```
## $`(Intercept)`
## [1] 5.007
##
## $Race
##   White   Black   Other
##  1.4451 -0.2226 -1.2225
##
## $Belief
##             Yes No or Undecided
##          0.7393         -0.7393
##
## $Race.Belief
##        Belief
## Race          Yes No or Undecided
##   White  0.008691       -0.008691
##   Black  0.037418       -0.037418
##   Other -0.046109        0.046109
```

```
life_loglm2<-loglm(~Race+Belief+Race*Belief,afterlife3,fitted=TRUE)   #or (accident_loglm1<-loglm(~1+2+
life_loglm2$param
```

```
## $`(Intercept)`
## [1] 5.007
##
## $Race
##   White   Black   Other
##  1.4451 -0.2226 -1.2225
##
## $Belief
##             Yes No or Undecided
##          0.7393         -0.7393
##
## $Race.Belief
##        Belief
## Race          Yes No or Undecided
##   White  0.008691       -0.008691
##   Black  0.037418       -0.037418
##   Other -0.046109        0.046109
```

Example: Alcohol, Cigarette, and Marijuana Use

```
data(marijuana2)
```

Table 7.4 Expected Value

```
# Eatablishment of logarithmic linear model
m_A_C_M=loglm(~Marijuana+Alcohol+Cigarettes,data=marijuana2,fitted=TRUE)
m_AC_M=loglm(~Alcohol*Cigarettes+Marijuana,data=marijuana2,fitted=TRUE)
m_AM_CM=loglm(~Alcohol*Marijuana+Cigarettes*Marijuana,data=marijuana2,fitted=TRUE)
m_AC_AM_CM=loglm(~Alcohol*Cigarettes+Alcohol*Marijuana+Cigarettes*Marijuana,
                 data=marijuana2,fitted=TRUE)
m_ACM=loglm(~Marijuana*Alcohol*Cigarettes,data=marijuana2,fitted=TRUE)
# Calculate the fitted values
m_A_C_M$fitted
```

```
## , , Marijuana = Yes
##
##        Cigarettes
## Alcohol   Yes     No
##     Yes 540.0 282.09
##     No   90.6  47.33
##
## , , Marijuana = No
##
##        Cigarettes
## Alcohol   Yes     No
##     Yes 740.2 386.70
##     No  124.2  64.88
```

```
m_AC_M$fitted
```

```
## , , Marijuana = Yes
##
##        Cigarettes
## Alcohol   Yes    No
##     Yes 611.2 210.9
##     No   19.4 118.5
##
## , , Marijuana = No
##
##        Cigarettes
## Alcohol   Yes    No
##     Yes 837.8 289.1
##     No   26.6 162.5
```

```
m_AM_CM$fitted
```

```
## , , Marijuana = Yes
##
##        Cigarettes
## Alcohol    Yes      No
##     Yes 909.24 45.7604
##     No    4.76  0.2396
##
## , , Marijuana = No
##
##        Cigarettes
## Alcohol   Yes    No
##     Yes 438.8 555.2
##     No  142.2 179.8
```

```
m_AC_AM_CM$fitted
```

```
## , , Marijuana = Yes
##
##        Cigarettes
## Alcohol     Yes     No
##     Yes 910.383 44.617
##     No    3.617  1.383
##
## , , Marijuana = No
##
##        Cigarettes
## Alcohol    Yes    No
##     Yes 538.62 455.4
##     No   42.38 279.6
```

```
m_ACM$fitted
```

```
## , , Marijuana = Yes
##
##        Cigarettes
## Alcohol Yes No
##     Yes 911 44
##     No    3  2
##
## , , Marijuana = No
```

```
##
##          Cigarettes
## Alcohol Yes  No
##     Yes 538 456
##     No   43 279
```

Table 7.5 Odds-Ratios

```
# Calculating conditional advantage ratio
(cond_AM_CM=(m_AM_CM$fitted[1,1,1]*m_AM_CM$fitted[2,2,1])/(m_AM_CM$fitted[1,2,1]*m_AM_CM$fitted[2,1,1])
```

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

```
# Calculating edge dominance Ratio
(mar_AM_CM=((m_AM_CM$fitted[1,1,1]+m_AM_CM$fitted[1,1,2])*(m_AM_CM$fitted[2,2,1]+m_AM_CM$fitted[2,2,2])
  ((m_AM_CM$fitted[1,2,1]+m_AM_CM$fitted[1,2,2])*(m_AM_CM$fitted[2,1,1]+m_AM_CM$fitted[2,1,2])))
```

```
## [1] 2.75
```

```
(cond_AC_M=(m_AC_M$fitted[1,1,1]*m_AC_M$fitted[2,2,1])/(m_AC_M$fitted[1,2,1]*m_AC_M$fitted[2,1,1]))
```

```
## [1] 17.7
```

```
(mar_AC_M=((m_AC_M$fitted[1,1,1]+m_AC_M$fitted[1,1,2])*(m_AC_M$fitted[2,2,1]+m_AC_M$fitted[2,2,2]))/
  ((m_AC_M$fitted[1,2,1]+m_AC_M$fitted[1,2,2])*(m_AC_M$fitted[2,1,1]+m_AC_M$fitted[2,1,2])))
```

```
## [1] 17.7
```

Table 7.6. Output for Fitting Loglinear Model to Table 7.3

```
marijuana2<-as.data.frame(marijuana2)
# Let the parameter of the second level of the variable be zero
marijuana2$Alcohol <- relevel(marijuana2$Alcohol,ref="No")
marijuana2$Cigarettes <- relevel(marijuana2$Cigarettes,ref="No")
marijuana2$Marijuana<- relevel(marijuana2$Marijuana,ref="No")
m=glm(marijuana2$Freq~marijuana2$Alcohol*marijuana2$Cigarettes+marijuana2$Alcohol*marijuana2$Marijuana+m
summary(m)
```

```
##
## Call:
## glm(formula = marijuana2$Freq ~ marijuana2$Alcohol * marijuana2$Cigarettes +
##     marijuana2$Alcohol * marijuana2$Marijuana + marijuana2$Cigarettes *
##     marijuana2$Marijuana, family = poisson)
##
## Deviance Residuals:
##       1        2        3        4        5        6
##  0.0204  -0.3343  -0.0926   0.4913  -0.0266   0.0945
##       7        8
##  0.0289  -0.0369
##
## Coefficients:
##                                                    Estimate
## (Intercept)                                          5.6334
## marijuana2$AlcoholYes                                0.4877
## marijuana2$CigarettesYes                            -1.8867
## marijuana2$MarijuanaYes                             -5.3090
## marijuana2$AlcoholYes:marijuana2$CigarettesYes       2.0545
## marijuana2$AlcoholYes:marijuana2$MarijuanaYes        2.9860
## marijuana2$CigarettesYes:marijuana2$MarijuanaYes     2.8479
##                                                    Std. Error
## (Intercept)                                            0.0597
## marijuana2$AlcoholYes                                  0.0758
## marijuana2$CigarettesYes                               0.1627
## marijuana2$MarijuanaYes                                0.4752
## marijuana2$AlcoholYes:marijuana2$CigarettesYes         0.1741
## marijuana2$AlcoholYes:marijuana2$MarijuanaYes          0.4647
## marijuana2$CigarettesYes:marijuana2$MarijuanaYes       0.1638
##                                                    z value
## (Intercept)                                          94.36
## marijuana2$AlcoholYes                                 6.44
## marijuana2$CigarettesYes                            -11.60
## marijuana2$MarijuanaYes                             -11.17
## marijuana2$AlcoholYes:marijuana2$CigarettesYes       11.80
## marijuana2$AlcoholYes:marijuana2$MarijuanaYes         6.43
## marijuana2$CigarettesYes:marijuana2$MarijuanaYes     17.38
##                                                    Pr(>|z|)
## (Intercept)                                         < 2e-16
## marijuana2$AlcoholYes                               1.2e-10
## marijuana2$CigarettesYes                            < 2e-16
## marijuana2$MarijuanaYes                             < 2e-16
## marijuana2$AlcoholYes:marijuana2$CigarettesYes      < 2e-16
## marijuana2$AlcoholYes:marijuana2$MarijuanaYes       1.3e-10
## marijuana2$CigarettesYes:marijuana2$MarijuanaYes    < 2e-16
##
## (Intercept)                                        ***
## marijuana2$AlcoholYes                              ***
## marijuana2$CigarettesYes                           ***
## marijuana2$MarijuanaYes                            ***
## marijuana2$AlcoholYes:marijuana2$CigarettesYes   ***
## marijuana2$AlcoholYes:marijuana2$MarijuanaYes    ***
## marijuana2$CigarettesYes:marijuana2$MarijuanaYes ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 2851.46098  on 7  degrees of freedom
## Residual deviance:    0.37399  on 1  degrees of freedom
## AIC: 63.42
##
## Number of Fisher Scoring iterations: 4
```

7.2 Inference for Loglinear Models

Table 7.7 Goodness of Fit Test

```
data(marijuana2)
# Calculating G^2,X^2,df and p value
(m_A_C_M=loglin(marijuana2,margin = list(1,2,3)))
```

```
## 2 iterations: deviation 0
```

```
## $lrt
## [1] 1286
##
## $pearson
## [1] 1411
##
## $df
## [1] 4
##
## $margin
## $margin[[1]]
## [1] "Alcohol"
##
## $margin[[2]]
## [1] "Cigarettes"
##
## $margin[[3]]
## [1] "Marijuana"
```

```
(m_A_CM=loglin(marijuana2,margin = list(1,c(2,3))))
```

```
## 2 iterations: deviation 7.105e-15
```

```
## $lrt
## [1] 534.2
##
## $pearson
## [1] 505.6
##
## $df
## [1] 3
##
## $margin
## $margin[[1]]
## [1] "Alcohol"
##
## $margin[[2]]
## [1] "Cigarettes" "Marijuana"
```

```
(m_C_AM=loglin(marijuana2,margin = list(2,c(1,3))))
```

```
## 2 iterations: deviation 0
```

```
## $lrt
## [1] 939.6
##
## $pearson
## [1] 824.2
##
## $df
## [1] 3
##
## $margin
## $margin[[1]]
## [1] "Cigarettes"
##
## $margin[[2]]
## [1] "Alcohol"   "Marijuana"
```

```
(m_M_AC=loglin(marijuana2,margin = list(3,c(1,2))))
```

```
## 2 iterations: deviation 0
```

```
## $lrt
## [1] 843.8
##
## $pearson
## [1] 704.9
##
## $df
## [1] 3
##
## $margin
## $margin[[1]]
## [1] "Marijuana"
##
## $margin[[2]]
## [1] "Alcohol"    "Cigarettes"
```

```
(m_AC_AM=loglin(marijuana2,margin = list(c(1,2),c(1,3))))
```

```
## 2 iterations: deviation 0
```

```
## $lrt
## [1] 497.4
##
## $pearson
## [1] 443.8
##
## $df
## [1] 2
##
## $margin
## $margin[[1]]
## [1] "Alcohol"    "Cigarettes"
##
## $margin[[2]]
## [1] "Alcohol"   "Marijuana"
```

```
(m_AC_CM=loglin(marijuana2,margin = list(c(1,2),c(2,3))))
```

```
## 2 iterations: deviation 0
```

```
## $lrt
## [1] 92.02
##
## $pearson
## [1] 80.81
##
```

```
## $df
## [1] 2
##
## $margin
## $margin[[1]]
## [1] "Alcohol"    "Cigarettes"
##
## $margin[[2]]
## [1] "Cigarettes" "Marijuana"
```

```
(m_AM_CM=loglin(marijuana2,margin = list(c(1,3),c(2,3))))
```

```
## 2 iterations: deviation 0
```

```
## $lrt
## [1] 187.8
##
## $pearson
## [1] 177.6
##
## $df
## [1] 2
##
## $margin
## $margin[[1]]
## [1] "Alcohol"   "Marijuana"
##
## $margin[[2]]
## [1] "Cigarettes" "Marijuana"
```

```
(m_AC_AM_CM=loglin(marijuana2,margin = list(c(1,2),c(1,3),c(2,3))))
```

```
## 5 iterations: deviation 0.03408
```

```
## $lrt
## [1] 0.374
##
## $pearson
## [1] 0.4011
##
## $df
## [1] 1
##
## $margin
## $margin[[1]]
## [1] "Alcohol"    "Cigarettes"
##
## $margin[[2]]
## [1] "Alcohol"   "Marijuana"
##
## $margin[[3]]
## [1] "Cigarettes" "Marijuana"
```

```
(m_ACM=loglin(marijuana2,margin = list(c(1,2,3))))
```

```
## 2 iterations: deviation 0
```

```
## $lrt
## [1] 0
##
## $pearson
## [1] 0
##
## $df
## [1] 0
##
## $margin
## $margin[[1]]
## [1] "Alcohol"    "Cigarettes" "Marijuana"
```

Table 7.8 Standardized Residuals

```
data(marijuana2)
library(boot)
fit1 <- glm(Freq ~ .+ Alcohol*Marijuana + Cigarettes*Marijuana, data=marijuana2,family=poisson)
glm.diag(fit1)
```

```
## $res
##       1       2       3       4       5       6       7
##   3.696  -3.709  -3.696   2.385  12.745 -13.794 -12.850
##       8
##  12.490
##
## $rd
##       1       2       3       4       5       6       7
##   3.694  -3.968  -3.720   2.290  12.363 -15.045 -13.217
##       8
##  11.837
##
```

```
## $rp
##       1       2       3       4       5       6       7
##   3.696  -3.696  -3.696   3.696  12.805 -12.805 -12.805
##       8
##  12.805
##
## $cook
##         1         2         3         4         5         6
## 9117.7755   45.4764  456.7199    0.1271  172.6362   37.4503
##         7         8
##  225.6383   54.6200
##
## $h
## [1] 0.99975 0.95233 0.99504 0.05288 0.86334 0.57815 0.89198
## [8] 0.66653
##
## $sd
## [1] 1
```

```
fit2 <- glm(Freq ~ .+ Alcohol*Marijuana + Cigarettes*Marijuana+Alcohol*Cigarettes, data=marijuana2,famil
glm.diag(fit2)
```

```
## $res
##       1       2       3       4       5       6       7
##  0.6333 -0.6385 -0.6334  0.6062 -0.6333  0.6333  0.6333
##       8
## -0.6333
##
## $rd
##       1       2       3       4       5       6       7
##  0.6333 -0.6527 -0.6348  0.5933 -0.6334  0.6318  0.6332
##       8
## -0.6336
##
## $rp
##       1       2       3       4       5       6       7
##  0.6333 -0.6333 -0.6333  0.6333 -0.6333  0.6333  0.6333
##       8
## -0.6333
##
## $cook
##        1        2        3        4        5        6
## 54.93488  0.16118  2.63780  0.02625 32.47814  2.50288
##        7        8
## 27.45036 16.83310
##
## $h
## [1] 0.9990 0.7377 0.9787 0.3142 0.9982 0.9776 0.9979 0.9966
##
## $sd
## [1] 1
```

Example:AutomobileAccidents and Seat Belts

```
data("accident_seatbelt2")
ftable(accident_seatbelt2)
```

```
##                          Injury    No   Yes
## Gender Location SeatBelt
## Female Urban    No               7287   996
##                 Yes             11587   759
##        Rural    No               3246   973
##                 Yes              6134   757
## Male   Urban    No              10381   812
##                 Yes             10969   380
##        Rural    No               6123  1084
##                 Yes              6693   513
```

Fit the seven log-linear models of Table 7.10.

```
fm_sbelt1<-loglm(~SeatBelt+Injury+Gender+Location,
                 accident_seatbelt2,
                 fitted=TRUE)
fm_sbelt2<-loglm(~SeatBelt+Injury+Gender+Location
                 +SeatBelt*Injury+SeatBelt*Gender
                 +SeatBelt*Location+Injury*Gender
                 +Injury*Location
                 +Gender*Location,
                 accident_seatbelt2,fitted=TRUE)
fm_sbelt3<-loglm(~SeatBelt+Injury+Gender+Location
                 +SeatBelt*Injury+SeatBelt*Gender
                 +SeatBelt*Location+Injury*Gender
                 +Injury*Location+Gender*Location
                 +SeatBelt*Injury*Gender+SeatBelt*Gender*Location
                 +SeatBelt*Injury*Location+Injury*Gender*Location,
                 accident_seatbelt2,fitted=TRUE)
```

```
fm_sbelt4<-loglm(~SeatBelt+Injury+Gender+Location
                 +SeatBelt*Injury+SeatBelt*Gender
                 +SeatBelt*Location+Injury*Gender
                 +Injury*Location+Gender*Location
                 +Injury*Gender*Location,
                 accident_seatbelt2,fitted=TRUE)
fm_sbelt5<-loglm(~SeatBelt+Injury+Gender+Location
                 +SeatBelt*Injury+SeatBelt*Gender
                 +SeatBelt*Location+Injury*Gender
                 +Injury*Location+Gender*Location
                 +SeatBelt*Injury*Gender,
                 accident_seatbelt2,fitted=TRUE)
fm_sbelt6<-loglm(~SeatBelt+Injury+Gender+Location
                 +SeatBelt*Injury+SeatBelt*Gender
                 +SeatBelt*Location+Injury*Gender
                 +Injury*Location+Gender*Location
                 +SeatBelt*Gender*Location,
                 accident_seatbelt2,fitted=TRUE)
fm_sbelt7<-loglm(~SeatBelt+Injury+Gender+Location
                 +SeatBelt*Injury+SeatBelt*Gender
                 +SeatBelt*Location+Injury*Gender
                 +Injury*Location+Gender*Location
                 +SeatBelt*Injury*Location,
                 accident_seatbelt2,fitted=TRUE)
```

Model (GI, GL, GS, IL, IS, LS) fitting,

```
ftable(fm_sbelt2$fitted)
```

```
##                          Injury      No     Yes
## Gender Location SeatBelt
## Female Urban    No               7166.4   993.0
##                 Yes             11748.3   721.3
##        Rural    No               3353.8   988.8
##                 Yes              5985.5   781.9
## Male   Urban    No              10471.5   845.1
##                 Yes             10837.8   387.6
##        Rural    No               6045.3  1038.1
##                 Yes              6811.4   518.2
```

Model (GLS, GI, IL, IS) fitting,

```
ftable(fm_sbelt6$fitted)
```

```
##                          Injury      No     Yes
## Gender Location SeatBelt
## Female Urban    No              7273.2  1009.8
##                 Yes            11632.6   713.4
##        Rural    No              3254.7   964.3
##                 Yes             6093.5   797.5
## Male   Urban    No             10358.9   834.1
##                 Yes            10959.2   389.8
##        Rural    No              6150.2  1056.8
##                 Yes             6697.6   508.4
```

Goodness-of-fit test for 7 log-linear models,

```
fm_sbelt1
```

```
## Call:
## loglm(formula = ~SeatBelt + Injury + Gender + Location, data = accident_seatbelt2,
##     fitted = TRUE)
##
## Statistics:
##                   X^2 df P(> X^2)
## Likelihood Ratio 2793 11        0
## Pearson          2758 11        0
```

```
fm_sbelt2
```

```
## Call:
## loglm(formula = ~SeatBelt + Injury + Gender + Location + SeatBelt *
##     Injury + SeatBelt * Gender + SeatBelt * Location + Injury *
##     Gender + Injury * Location + Gender * Location, data = accident_seatbelt2,
##     fitted = TRUE)
##
## Statistics:
##                    X^2 df  P(> X^2)
## Likelihood Ratio 23.35  5 0.0002892
## Pearson          23.38  5 0.0002861
```

```
fm_sbelt3
```

```
## Call:
## loglm(formula = ~SeatBelt + Injury + Gender + Location + SeatBelt *
##     Injury + SeatBelt * Gender + SeatBelt * Location + Injury *
##     Gender + Injury * Location + Gender * Location + SeatBelt *
##     Injury * Gender + SeatBelt * Gender * Location + SeatBelt *
##     Injury * Location + Injury * Gender * Location, data = accident_seatbelt2,
##     fitted = TRUE)
##
```

```
## Statistics:
##                     X^2 df P(> X^2)
## Likelihood Ratio 1.325  1   0.2496
## Pearson          1.325  1   0.2498
```

```
fm_sbelt4
```

```
## Call:
## loglm(formula = ~SeatBelt + Injury + Gender + Location + SeatBelt *
##     Injury + SeatBelt * Gender + SeatBelt * Location + Injury *
##     Gender + Injury * Location + Gender * Location + Injury *
##     Gender * Location, data = accident_seatbelt2, fitted = TRUE)
##
## Statistics:
##                     X^2 df  P(> X^2)
## Likelihood Ratio 18.57  4 0.0009548
## Pearson          18.54  4 0.0009679
```

```
fm_sbelt5
```

```
## Call:
## loglm(formula = ~SeatBelt + Injury + Gender + Location + SeatBelt *
##     Injury + SeatBelt * Gender + SeatBelt * Location + Injury *
##     Gender + Injury * Location + Gender * Location + SeatBelt *
##     Injury * Gender, data = accident_seatbelt2, fitted = TRUE)
##
## Statistics:
##                     X^2 df  P(> X^2)
## Likelihood Ratio 22.85  4 0.0001359
## Pearson          22.82  4 0.0001372
```

```
fm_sbelt6
```

```
## Call:
## loglm(formula = ~SeatBelt + Injury + Gender + Location + SeatBelt *
##     Injury + SeatBelt * Gender + SeatBelt * Location + Injury *
##     Gender + Injury * Location + Gender * Location + SeatBelt *
##     Gender * Location, data = accident_seatbelt2, fitted = TRUE)
##
## Statistics:
##                     X^2 df P(> X^2)
## Likelihood Ratio 7.464  4   0.1133
## Pearson          7.487  4   0.1123
```

```
fm_sbelt7
```

```
## Call:
## loglm(formula = ~SeatBelt + Injury + Gender + Location + SeatBelt *
##     Injury + SeatBelt * Gender + SeatBelt * Location + Injury *
##     Gender + Injury * Location + Gender * Location + SeatBelt *
##     Injury * Location, data = accident_seatbelt2, fitted = TRUE)
##
## Statistics:
##                       X^2 df  P(> X^2)
## Likelihood Ratio 20.63  4 0.0003743
## Pearson          20.61  4 0.0003778
```

Estimation of conditional odds ratios for two log-linear models (Model1–(GI, GL, GS, IL, IS, LS), Model2–(GLS, GI, IL, IS))

```
Oddratio<-c("GI","IL","IS","GL(S=No)","GL(S=Yes)","GS(L=Urban)","GS(L=Rural)","LS(G=Female)","LS(G=Male
GI1<-oddsratio(fm_sbelt2$fitted[,1,1,])  # or oddsratio(fm_sbelt2$fitted[,2,2,])
IL1<-oddsratio(fm_sbelt2$fitted[1,,1,])
IS1<-oddsratio(fm_sbelt2$fitted[1,1,,])
GLSN1<-oddsratio(fm_sbelt2$fitted[,,1,1])
GLSY1<-oddsratio(fm_sbelt2$fitted[,,2,1])
GSLU1<-oddsratio(fm_sbelt2$fitted[,1,,1])
GSLR1<-oddsratio(fm_sbelt2$fitted[,2,,1])
LSGF1<-oddsratio(fm_sbelt2$fitted[1,,,1])
LSGM1<-oddsratio(fm_sbelt2$fitted[2,,,1])
Model1<-c(GI1,IL1,IS1,GLSN1,GLSY1,GSLU1,GSLR1,LSGF1,LSGM1)
GI2<-oddsratio(fm_sbelt6$fitted[,1,1,])
IL2<-oddsratio(fm_sbelt6$fitted[1,,1,])
IS2<-oddsratio(fm_sbelt6$fitted[1,1,,])
GLSN2<-oddsratio(fm_sbelt6$fitted[,,1,1])
GLSY2<-oddsratio(fm_sbelt6$fitted[,,2,1])
GSLU2<-oddsratio(fm_sbelt6$fitted[,1,,1])
GSLR2<-oddsratio(fm_sbelt6$fitted[,2,,1])
LSGF2<-oddsratio(fm_sbelt6$fitted[1,,,1])
LSGM2<-oddsratio(fm_sbelt6$fitted[2,,,1])
Model2<-c(GI2,IL2,IS2,GLSN2,GLSY2,GSLU2,GSLR2,LSGF2,LSGM2)
cbind(Oddratio,Model1,Model2)
```

```
##              Oddratio      Model1    Model2
## coefficients "GI"          -0.5405   -0.5448
## dimnames     "IL"          List,2    List,2
## dim          "IS"          Integer,2 Integer,2
## vcov         "GL(S=No)"    0.002425  0.002423
## contrasts    "GL(S=Yes)"    Integer,4 Integer,4
## log          "GS(L=Urban)"  TRUE      TRUE
## coefficients "GS(L=Rural)"  0.755     0.7581
## dimnames     "LS(G=Female)" List,2    List,2
## dim          "LS(G=Male)"   Integer,2 Integer,2
## vcov         "GI"           0.002456  0.002472
## contrasts    "IL"           Integer,4 Integer,4
## log          "IS"           TRUE      TRUE
## coefficients "GL(S=No)"     -0.814    -0.8171
## dimnames     "GL(S=Yes)"    List,2    List,2
## dim          "GS(L=Urban)"  Integer,2 Integer,2
## vcov         "GS(L=Rural)"  0.002618  0.002616
## contrasts    "LS(G=Female)" Integer,4 Integer,4
## log          "LS(G=Male)"   TRUE      TRUE
## coefficients "GI"           0.2099    0.2827
## dimnames     "IL"           List,2    List,2
## dim          "IS"           Integer,2 Integer,2
## vcov         "GL(S=No)"     0.0006986 0.0007039
## contrasts    "GL(S=Yes)"    Integer,4 Integer,4
## log          "GS(L=Urban)"  TRUE      TRUE
## coefficients "GS(L=Rural)"  0.2099    0.1542
## dimnames     "LS(G=Female)" List,2    List,2
## dim          "LS(G=Male)"   Integer,2 Integer,2
## vcov         "GI"           0.0004913 0.0004906
## contrasts    "IL"           Integer,4 Integer,4
## log          "IS"           TRUE      TRUE
## coefficients "GL(S=No)"     -0.4599   -0.4133
## dimnames     "GL(S=Yes)"    List,2    List,2
## dim          "GS(L=Urban)"  Integer,2 Integer,2
## vcov         "GS(L=Rural)"  0.0004124 0.0004112
## contrasts    "LS(G=Female)" Integer,4 Integer,4
## log          "LS(G=Male)"   TRUE      TRUE
## coefficients "GI"           -0.4599   -0.5419
## dimnames     "IL"           List,2    List,2
## dim          "IS"           Integer,2 Integer,2
## vcov         "GL(S=No)"     0.0007775 0.0007833
## contrasts    "GL(S=Yes)"    Integer,4 Integer,4
## log          "GS(L=Urban)"  TRUE      TRUE
## coefficients "GS(L=Rural)"  0.08493   0.1575
## dimnames     "LS(G=Female)" List,2    List,2
## dim          "LS(G=Male)"   Integer,2 Integer,2
## vcov         "GI"           0.0006899 0.0006948
## contrasts    "IL"          Integer,4 Integer,4
## log          "IS"          TRUE      TRUE
## coefficients "GL(S=No)"    0.08493   0.02894
## dimnames     "GL(S=Yes)"   List,2    List,2
## dim          "GS(L=Urban)" Integer,2 Integer,2
## vcov         "GS(L=Rural)" 5e-04     0.0004997
## contrasts    "LS(G=Female)" Integer,4 Integer,4
## log          "LS(G=Male)"  TRUE      TRUE
```

Large Samples and Statistical Versus Practical Significance

Different indicators of models (GI, GL, GS, IL, IS, LS) and (GLS, GI, IL, IS)

```
sum(abs(fm_sbelt2$fitted-fm_sbelt2$frequencies))/(2*68694)
```

```
## [1] 0.008219
```

```
sum(abs(fm_sbelt6$fitted-fm_sbelt6$frequencies))/(2*68694)
```

```
## [1] 0.002507
```

7.3 The Loglinear–Logistic Connection

7.4 Independence Graphs and Collapsibility

Example: Model Building for Student Drug Use

```
data("marijuana")
mi1<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race+Gender*Race,marijuana)
mi2<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
           +Cigarettes*Alcohol+Cigarettes*Marijuana
           +Cigarettes*Gender+Cigarettes*Race
           +Alcohol*Marijuana+Alcohol*Gender
           +Alcohol*Race+Marijuana*Gender
           +Marijuana*Race+Gender*Race,marijuana)
mi3<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
           +Cigarettes*Alcohol+Cigarettes*Marijuana
           +Cigarettes*Gender+Cigarettes*Race
           +Alcohol*Marijuana+Alcohol*Gender
           +Alcohol*Race+Marijuana*Gender
           +Marijuana*Race+Gender*Race
           +Cigarettes*Alcohol*Marijuana
           +Cigarettes*Alcohol*Gender
           +Cigarettes*Alcohol*Race
           +Cigarettes*Marijuana*Gender
           +Cigarettes*Marijuana*Race
           +Cigarettes*Gender*Race
           +Alcohol*Marijuana*Gender
           +Alcohol*Marijuana*Race+Alcohol*Gender*Race
           +Marijuana*Gender*Race,marijuana)
mi4a<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Marijuana+Cigarettes*Gender
            +Cigarettes*Race+Alcohol*Marijuana
            +Alcohol*Gender+Alcohol*Race
            +Marijuana*Gender+Marijuana*Race+Gender*Race,marijuana)
mi4b<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Alcohol+Cigarettes*Marijuana
            +Cigarettes*Gender+Cigarettes*Race
            +Alcohol*Gender+Alcohol*Race+Marijuana*Gender
            +Marijuana*Race+Gender*Race,marijuana)
mi4c<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Alcohol+Cigarettes*Gender
            +Cigarettes*Race+Alcohol*Marijuana
            +Alcohol*Gender+Alcohol*Race+Marijuana*Gender
            +Marijuana*Race+Gender*Race,marijuana)
mi4d<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Alcohol+Cigarettes*Marijuana
            +Cigarettes*Gender+Cigarettes*Race
            +Alcohol*Marijuana+Alcohol*Race+Marijuana*Gender
            +Marijuana*Race+Gender*Race,marijuana)
mi4e<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Alcohol+Cigarettes*Marijuana
            +Cigarettes*Gender+Cigarettes*Race
            +Alcohol*Marijuana+Alcohol*Gender
            +Marijuana*Gender+Marijuana*Race+Gender*Race,marijuana)
mi4f<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Alcohol+Cigarettes*Marijuana
            +Cigarettes*Race+Alcohol*Marijuana
            +Alcohol*Gender+Alcohol*Race+Marijuana*Gender
            +Marijuana*Race+Gender*Race,marijuana)
mi4g<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Alcohol+Cigarettes*Marijuana
            +Cigarettes*Gender+Alcohol*Marijuana
            +Alcohol*Gender+Alcohol*Race+Marijuana*Gender
            +Marijuana*Race+Gender*Race,marijuana)
mi4h<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Alcohol+Cigarettes*Marijuana
            +Cigarettes*Gender+Cigarettes*Race
            +Alcohol*Marijuana+Alcohol*Gender+Alcohol*Race
            +Marijuana*Race+Gender*Race,marijuana)
mi4i<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
            +Cigarettes*Alcohol+Cigarettes*Marijuana
            +Cigarettes*Gender+Cigarettes*Race
            +Alcohol*Marijuana+Alcohol*Gender+Alcohol*Race
            +Marijuana*Gender+Gender*Race,marijuana)
mi5<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
           +Cigarettes*Alcohol+Cigarettes*Marijuana
           +Alcohol*Marijuana+Alcohol*Gender+Alcohol*Race
           +Marijuana*Gender+Marijuana*Race+Gender*Race,marijuana)
mi6<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
           +Cigarettes*Alcohol+Cigarettes*Marijuana
           +Alcohol*Marijuana+Alcohol*Gender+Alcohol*Race
           +Marijuana*Gender+Gender*Race,marijuana)
mi7<-loglm(~Cigarettes+Alcohol+Marijuana+Gender+Race
           +Cigarettes*Alcohol+Cigarettes*Marijuana
           +Alcohol*Marijuana+Alcohol*Race+Marijuana*Gender
           +Gender*Race,marijuana)
model<-c("    +GR","   ","    ",
         "(2)-AC","(2)-AM","(2)-CM","(2)-AG","(2)-AR",
         "(2)-CG","(2)-CR","(2)-GM","(2)-MR",
         "(AC,AM,CM,AG,AR,GM,GR,MR)","(AC,AM,CM,AG,AR,GM,GR)",
         "(AC,AM,CM,AR,GM,GR)")
G2<-c(mi1$lrt,mi2$lrt,mi3$lrt,mi4a$lrt,mi4b$lrt,
      mi4c$lrt,mi4d$lrt,mi4e$lrt,mi4f$lrt,mi4g$lrt,
      mi4h$lrt,mi4i$lrt,mi5$lrt,mi6$lrt,mi7$lrt)
df<-c(mi1$df,mi2$df,mi3$df,mi4a$df,mi4b$df,
      mi4c$df,mi4d$df,mi4e$df,mi4f$df,mi4g$df,
      mi4h$df,mi4i$df,mi5$df,mi6$df,mi7$df)
result_mi<-cbind(model,G2,df)
result_mi
```

```
##       model                       G2                 df
##  [1,] "  +GR"               "1325.14076131126" "25"
##  [2,] "   "                 "15.3403512047516" "16"
##  [3,] "    "              "5.27205597739754" "6"
##  [4,] "(2)-AC"                   "201.199314356916" "17"
##  [5,] "(2)-AM"                   "106.958003256214" "17"
##  [6,] "(2)-CM"                   "513.472183973222" "17"
##  [7,] "(2)-AG"                   "18.7169544989871" "17"
##  [8,] "(2)-AR"                   "20.3208672411299" "17"
##  [9,] "(2)-CG"                   "16.3171920778461" "17"
## [10,] "(2)-CR"                   "15.7834781637697" "17"
## [11,] "(2)-GM"                   "25.1610149959952" "17"
## [12,] "(2)-MR"                   "18.9289431100161" "17"
## [13,] "(AC,AM,CM,AG,AR,GM,GR,MR)" "16.735039750754"  "18"
## [14,] "(AC,AM,CM,AG,AR,GM,GR)"    "19.9085873883873" "19"
## [15,] "(AC,AM,CM,AR,GM,GR)"       "25.1683603785765" "20"
```

7.5 Modeling Ordinal Associations

```
find_data_by_title("sex")
```

```
## [1] "premarital_sex1" "premarital_sex2" "sexual_behavior"
## [4] "teen_sex"
```

```
data("premarital_sex2")
# The goodness-of-fit statistics of the loglinear model of independence, (X,Y)
(X2_ind <- independent_test_of_table(premarital_sex2,method = "X2")$statistic)
```

```
## [1] 128.7
```

```
(G2_ind <- independent_test_of_table(premarital_sex2,method = "G2")$statistic)
```

```
## [1] 127.7
```

```
# Calculate the fitted values and standardized residuals of the dependence model
(E <- round(chisq.test(premarital_sex2)$expected,digits = 1))
```

```
##                         BirthControl
## PremaritalSex            Strongly Disagree Disagree Agree
##   Always wrong                        42.4     51.2  86.4
##   Almost always wrong                 16.0     19.3  32.5
##   Wrong only sometimes                30.0     36.3  61.2
##   Not wrong at all                    70.6     85.2 143.8
##                         BirthControl
## PremaritalSex            Strongly Agree
##   Always wrong                     67.0
##   Almost always wrong              25.2
##   Wrong only sometimes             47.4
##   Not wrong at all                111.4
```

```r
(R <- round(chisq.test(premarital_sex2)$stdres,digits = 1))
```

```
##                         BirthControl
## PremaritalSex           Strongly Disagree Disagree Agree
##   Always wrong                        7.6      3.1  -4.1
##   Almost always wrong                 2.3      1.8  -0.8
##   Wrong only sometimes               -2.7      1.0   2.2
##   Not wrong at all                   -6.1     -4.6   2.4
##                         BirthControl
## PremaritalSex           Strongly Agree
##   Always wrong                    -4.8
##   Almost always wrong             -2.8
##   Wrong only sometimes            -1.0
##   Not wrong at all                 6.8
```

Example: Sex Opinions

```r
data("premarital_sex2")
# Conduct the linear-by-linear association model with u,v={1,2,3,4}
u <-rep(c(1,2,3,4),4)
v <- c(1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4)
LL <- glm(Freq ~ PremaritalSex + BirthControl + u:v, data=premarital_sex2,family=poisson)
summary(LL)
```

```
##
## Call:
## glm(formula = Freq ~ PremaritalSex + BirthControl + u:v, family = poisson,
##     data = premarital_sex2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3583  -0.9161   0.0797   0.6165   1.5762
##
## Coefficients:
##                                    Estimate Std. Error
## (Intercept)                          4.1068     0.0895
## PremaritalSexAlmost always wrong    -1.6460     0.1347
## PremaritalSexWrong only sometimes   -1.7700     0.1646
## PremaritalSexNot wrong at all       -1.7537     0.2343
## BirthControlDisagree                -0.4641     0.1195
## BirthControlAgree                   -0.7245     0.1620
## BirthControlStrongly Agree          -1.8797     0.2491
## u:v                                 0.2858     0.0282
##                                  z value Pr(>|z|)
## (Intercept)                        45.88  < 2e-16 ***
## PremaritalSexAlmost always wrong  -12.22  < 2e-16 ***
## PremaritalSexWrong only sometimes -10.75  < 2e-16 ***
## PremaritalSexNot wrong at all      -7.48  7.2e-14 ***
## BirthControlDisagree               -3.88    1e-04 ***
## BirthControlAgree                  -4.47  7.7e-06 ***
## BirthControlStrongly Agree         -7.55  4.5e-14 ***
## u:v                                10.12  < 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: 431.078  on 15  degrees of freedom
## Residual deviance:  11.534  on  8  degrees of freedom
## AIC: 118.2
##
## Number of Fisher Scoring iterations: 4
```

```
# Calculate the fitted values based on the model in Table 7.15
(E.fit <- matrix(LL$fitted.values,ncol=4))
```

```
##       [,1]  [,2]   [,3]   [,4]
## [1,] 80.86 67.65  69.40  29.09
## [2,] 20.75 23.11  31.54  17.60
## [3,] 24.39 36.15  65.68  48.77
## [4,] 33.00 65.09 157.38 155.53
```

```
# The goodness-of-fit statistics of the model

(G2_LL <- LL$deviance)
```

```
## [1] 11.53
```

```
(X2_LL <- chisqstat(LL))
```

```
## [1] 11.51
```

```
# Conduct the linear-by-linear association models with different u,v scores
u1 <- rep(c(-1.5, -0.5,  0.5,  1.5),4)
LL1 <- glm(Freq ~ PremaritalSex + BirthControl + u1:v, data=premarital_sex2,family=poisson)
summary(LL1)
```

```
##
## Call:
## glm(formula = Freq ~ PremaritalSex + BirthControl + u1:v, family = poisson,
##     data = premarital_sex2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3583  -0.9161   0.0797   0.6165   1.5762
##
## Coefficients:
##                                   Estimate Std. Error
## (Intercept)                         4.8214     0.1226
## PremaritalSexAlmost always wrong   -1.6460     0.1347
## PremaritalSexWrong only sometimes  -1.7700     0.1646
## PremaritalSexNot wrong at all      -1.7537     0.2343
## BirthControlDisagree                0.2505     0.1097
## BirthControlAgree                   0.7047     0.1010
## BirthControlStrongly Agree          0.2641     0.1081
## u1:v                                0.2858     0.0282
##                                   z value Pr(>|z|)
## (Intercept)                         39.32  < 2e-16 ***
## PremaritalSexAlmost always wrong   -12.22  < 2e-16 ***
## PremaritalSexWrong only sometimes  -10.75  < 2e-16 ***
## PremaritalSexNot wrong at all       -7.48  7.2e-14 ***
## BirthControlDisagree                 2.28    0.022 *
## BirthControlAgree                    6.97  3.1e-12 ***
## BirthControlStrongly Agree           2.44    0.015 *
## u1:v                                10.12  < 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: 431.078  on 15  degrees of freedom
## Residual deviance:  11.534  on  8  degrees of freedom
## AIC: 118.2
##
## Number of Fisher Scoring iterations: 4
```

```
u2 <- rep(c(2, 4,  6,  8),4)
LL2 <- glm(Freq ~ PremaritalSex + BirthControl + u2:v, data=premarital_sex2,family=poisson)
```

```
LL2$deviance
```

```
## [1] 11.53
```

```
u3 <- rep(c(1, 2, 4, 5),4)
v3 <- c(1,1,1,1,2,2,2,2,4,4,4,4,5,5,5,5)
LL3 <- glm(Freq ~ PremaritalSex + BirthControl + u3:v3, data=premarital_sex2,family=poisson)
LL3$deviance
```

```
## [1] 8.845
```

Ordinal Tests of Independence

```
# The likelihood-ratio test statistic
(G2_XY_LL <- G2_ind-G2_LL)
```

```
## [1] 116.1
```

```
1-pnorm(G2_XY_LL)
```

```
## [1] 0
```

```
# The Wald statistic
(z2 <- (summary(LL)$coefficients[8,1]/summary(LL)$coefficients[8,2])^2)
```

```
## [1] 102.5
```

```
1-pnorm(z2)
```

```
## [1] 0
```

APPENDIX

Appendix A. INTRODUCTION TO THE USE OF THE R PACKAGE

A.1 Preparation

There are several basic R packages you need to install fisrt.

- `cdabookdb` - containing the datasets in the textbook, including dataset for the Examples and also the Exercises. [Required]
- `cdabookfunc` - some useful functions to conduct statistical inferences. [Required]
- `icda` - supplementary R package for this book. [Optional]
- `elrm` - conduct exact-like conditional logistic regression inferences (Chapter 5). [Required]
- `logistiX` - conduct exact conditional logistic regression inferences(Chapter 5). [Required]

A.2 Installation

Way (i): Extract the compiled R package to the library directory of R (you can run `.libPaths()` view in R).

Way (ii): [Recommend]

- If your computer has a Windows operating system, then you can download the `.zip` file. And if your computer has a Mac operating system, we suggest you to download the `.tar.gz` file.
- Then Find `Tools` in the menu bar.
- Choose the fisrt bottem `Install Packages`.
- Install from `Package Archive File`.
- Choose the `.zip`(Windows) or `.tar.gz`(Mac or Win) file to install.
- Done!

Way (iii): Run

```
install.packages(fpath, repos = NULL, type = "source")
```

where `fpath` is the path storing the `.zip` or `.tar.gz` file.

In addition, the package used in this code document has been listed as a package for `cdabookdb`. To install these suggestion packages you can specify parameters when installation.

You can get the suggestion package and install it like this,

```r
suggested_pkgs <- packageDescription("cdabookdb")$Suggests
suggested_pkgs <- strsplit(suggested_pkgs, ",\\s*")[[1]]
suggested_pkgs
```

```
##  [1] "knitr"             "rmarkdown"
##  [3] "Fahrmeir"          "binom"
##  [5] "dplyr"             "MASS"
##  [7] "pROC"              "ResourceSelection"
##  [9] "ROCR"              "tidyr"
## [11] "VGAM"
```

```r
# Install if not installed
lapply(suggested_pkgs, function(pkg) {
  if (system.file(package = pkg) == '') install.packages(pkg)
})
```

A.3 Instructions for Use

You can use `data(package = "cdabookdb")` to view the datasets contained in the package and their descriptions.

```r
library(cdabookdb)
data(package = "cdabookdb")$results[, 3]
```

```
##  [1] "AIDS_treatment"
##  [2] "AZT"
##  [3] "Behaviors_to_help_environment"
##  [4] "Choice_of_Coffee"
##  [5] "Diagnoses_of_Carcinoma"
##  [6] "Jour_cite"
##  [7] "MBtest1"
##  [8] "MBtest2"
##  [9] "MBtest3"
## [10] "MI_Pairs"
## [11] "Migraine_treatment"
## [12] "UCBAdmissions"
## [13] "UFAdmissions"
## [14] "accident_seatbelt1"
## [15] "accident_seatbelt2"
## [16] "accident_seatbelt3"
## [17] "afterlife1"
## [18] "afterlife2"
## [19] "afterlife3"
## [20] "albumin"
## [21] "alligators1"
## [22] "alligators2"
## [23] "aspirin"
## [24] "aspr_heart"
## [25] "athlete_graduate"
## [26] "birth_control"
## [27] "blood_pressure"
## [28] "cancer_remission"
## [29] "chip_imperfection"
## [30] "cholesterol"
## [31] "credit_score"
## [32] "creditcard"
## [33] "deathpenalty1"
## [34] "deathpenalty2"
## [35] "edu_aspiration"
## [36] "environment_crisis"
## [37] "environment_pro"
## [38] "environmental_protection"
## [39] "football_arrest"
## [40] "gender_party"
## [41] "government_spending"
## [42] "happiness1"
## [43] "happiness2"
## [44] "happiness3"
## [45] "horseshoecrabs"
## [46] "ideology"
## [47] "impairment"
## [48] "incontinent"
## [49] "job_satisfaction1"
## [50] "job_satisfaction2"
## [51] "job_satisfaction3"
## [52] "kyphosis_age"
## [53] "larynx_cancer"
## [54] "lungcancer_treatment"
## [55] "malformation"
## [56] "marijuana"
## [57] "marijuana2"
## [58] "marital_happiness"
## [59] "merit_pay_race"
## [60] "missing_persons"
## [61] "mutiple_sclerosis"
## [62] "osteosarcoma"
## [63] "prednisolone"
## [64] "premarital_sex1"
## [65] "premarital_sex2"
## [66] "promotion_race"
## [67] "psych_diag_drugs"
## [68] "rabbit_penicillin"
## [69] "race_party"
## [70] "religious_belief"
## [71] "religious_belief_change"
## [72] "residence"
## [73] "sexual_behavior"
## [74] "smoking_cd"
## [75] "smoking_lungcancer"
## [76] "smoking_lungcancer_cn"
## [77] "smoking_mi"
## [78] "snoring_heartdisease"
## [79] "social_survey"
## [80] "teen_sex"
## [81] "teenager_crime"
## [82] "temperature_distress"
## [83] "tennis"
## [84] "tennis_match"
## [85] "throat"
## [86] "toxicity"
## [87] "traincollisions"
## [88] "treatment1"
## [89] "treatment2"
## [90] "treatment3"
## [91] "white_black_acceptance"
```

Data sets can be introduced using `data(DATANAME)`.

In addition, the `cdabookfunc` package contains several useful functions.

```
sort(getNamespaceExports("cdabookfunc"))
```

```
##  [1] "binom_inference"            "binom_mid_pvalue"
##  [3] "Binomial_To_Binary"         "chisqstat"
##  [5] "cmh.test"                   "dfbetas_logit_sas"
##  [7] "diff_prop"                "exact_test_for_22K"
##  [9] "find_data_by_title"       "find_data_by_var"
## [11] "independent_test_of_table" "influence_logit_sas"
## [13] "oddsratio"                "r_compute"
## [15] "samplesize_logit"         "samplesize_multilogit"
## [17] "samplesize_prop"          "stdres"
```

The use of the function is shown in the following table.

Function Name	Description	Refering Section
find_data_by_title	Find the data set based on the title of the data	NULL
find_data_by_var	Find the data set based on the variable name	NULL
binom_inference	Inference of binomial distribution	1.4.2
binom_mid_pvalue	Binomial Mid-P value	1.4.5
diff_prop	Calculate the difference of proportions	2.2.1
oddsratio	Calculate the odds ratio	2.3.1
independent_test_of_table	Three independence test method for contingency tables	2.4-2.5
cmh.test	Conduct the CMH-test	4.3.4
dfbetas_logit_sas	Calculation of logistic regression of dfbetas by SAS	5.2.7
influence_logit_sas	Logistic regression diagnosis using SAS	5.2.7
samplesize_prop	Calculate the sample size required to compare two props	5.5.1
Binomial_To_Binary	Transfers binomial responses to binary responses	NULL
chisqstat	Compute Pearson's chi-square goodness-of-fit statistic	CH7
exact_test_for_22K	Conducts exact test for 22K contingency tables	5.4
r_compute	Computes Correlation r between Two Ordinal Variables	6.3
samplesize_logit	Calculates sample size required in logistic regression	5.5
samplesize_multilogit	Sample size required in multiple logistic regression	5.5
stdres	Residuals for Cells in a Contingency Table	2.4.5

The first two functions are used to find the data set needed in a large data set from `cdabookdb` (default, or a specified package or all installed packages, etc., to see the help information of the function). Starting with the third function is used to facilitate the implementation of the code results on the book.

Appendix B. LIST FOR DATA IN THE TEXTBOOK

B.1 Data for Examples in the Front

The following table is the case data set used in the text of the textbook (can be found in `cdabookdb`).

Section	Name of the Case	Data
2.1	Belief in Afterlife	afterlife1
2.2	Aspirin Use and Myocardial Infarction	aspirin
2.3	Aspirin Use and Myocardial Infarction	aspirin
2.3	Smoking Status and Myocardial Infarction	smoking_mi
2.4	Party Identification and Gender	gender_party
2.5	Infant Malformation and Alcohol Consumption	malformation
2.7	Death Penalty Verdict by Defendant and Victims	deathpenalty1
2.7	Response drug treatment and clinic	treatment1
3.2	Snoring and Heart Disease	snoring_heartdisease
3.3	Horseshoe Crabs(Poisson GLM)	horseshoecrabs
3.3	Horseshoe Crabs(Negative Binomial GLM)	horseshoecrabs
3.3	Collisions Involving Trains in Great Britain	traincollisions
3.4	Snoring and Heart Disease	snoring_heartdisease
4.1	Horseshoe Crabs(logistic regression)	horseshoecrabs
4.3	AIDS Symptoms by AZT Use and Race	AZT
4.4	Horseshoe Crabs Revisited(Multiple logistic)	horseshoecrabs
5.1	Horseshoe Crabs Revisited(Model Selection)	horseshoecrabs
5.1	Horseshoe Crabs Revisited(Predicted power)	horseshoecrabs
5.2	Horseshoe Crabs Revisited(LR test)	horseshoecrabs
5.2	AIDS Symptoms by AZT Use and Race	AZT
5.2	Horseshoe Crabs Revisited(HM test)	horseshoecrabs
5.2	Admission to Graduate School at Florida	UFAdmissions
5.2	Heart Disease Data	blood_pressure
5.3	Clinical Trial Relating Treatment	treatment3

5.4	Promotion Decisions by Race and by Month	promotion_race
6.1	Alligator Size and Primary Food Choice	alligators1
6.1	Belief in Afterlife by Gender and Race	afterlife2
6.2	Political Ideology by Gender and Political Party	ideology
6.2	Mental Impairment by SES and Life Events	impairment
6.3	Political Ideology Revisited	ideology
6.3	Pregnant Mice in Developmental Toxicity Study	toxicity
6.4	Job Satisfaction and Income	job_satisfaction2
7.1	Social Survey on belief in life after death	afterlife3
7.1	Alcohol Cigarette and Marijuana Use	marijuana2
7.5	Premarital Sex and Teenage Birth Control	premarital_sex2
8.1	Opinions Relating to Environment	environmental_protection
8.2	Previous Diagnoses of Diabetes for MI	MI_Pairs
8.3	Choice of Decaffeinated Coffee	Choice_of_Coffee
8.3	Behaviors on Helping Environment	Behaviors_to_help_environment
8.5	Diagnoses of Carcinoma	Diagnoses_of_Carcinoma
8.6	2004–2005 Tennis Matches for Men Players	tennis_match

B.2 Data for Exercises Problems

The following table is the data set used in the exercises of the textbook (can be found in `cdabookdb`).

Problem	Data
2.16	smoking_lungcancer
2.18	happiness1
2.19	race_party
2.21	teenager_crime
2.22	psych_diag_drugs
2.23	religious_belief
2.27	edu_aspiration
2.29	prednisolone
2.3	larynx_cancer
2.33	deathpenalty2
3.3	malformation
3.4	malformation
3.5	snoring_heartdisease
3.6	snoring_heartdisease
3.7	horseshoecrabs
3.8	horseshoecrabs
3.9	creditcard
3.1	cancer_remission

3.11	chip_imperfection
3.12	chip_imperfection
3.13	horseshoecrabs
3.14	horseshoecrabs
3.18	football_arrest
3.19	traincollisions
3.2	smoking_cd
4.1	cancer_remission
4.2	cancer_remission
4.4	snoring_heartdisease
4.5	temperature_distress
4.6	creditcard
4.7	kyphosis_age
4.8	horseshoecrabs
4.12	deathpenalty2
4.13	deathpenalty2
4.14	AZT
4.15	merit_pay_race
4.16	MBtest
4.17	MBtest
4.2	treatment2
4.22	horseshoecrabs
4.24	throat
4.25	horseshoecrabs
4.26	horseshoecrabs
4.27	horseshoecrabs
4.29	teen_sex
4.3	athlete_graduate
4.31	marijuana
4.32	albumin
4.33	job_satisfaction_survey
4.37	deathpenalty1
5.1	horseshoecrabs
5.2	horseshoecrabs
5.3	horseshoecrabs
5.4	MBtest1
5.6	MBtest1
5.7	MBtest2
5.9	cancer_remission
5.1	horseshoecrabs
5.11	horseshoecrabs

5.12	premarital_sex1
5.13	credit_score
5.15	missing_persons
5.17	deathpenalty1
5.18	smoking_lungcancer_cn
5.19	UCBAdmissions
5.2	malformation
5.21	malformation
5.23	rabbit_penicillin
5.24	rabbit_penicillin
5.25	osteosarcoma
5.26	incontinent
5.29	horseshoecrabs
6.2	alligators1
6.3	alligators2
6.4	afterlife2
6.6	marital_happiness
6.7	marital_happiness
6.8	lungcancer_treatment
6.1	impairment
6.11	job_satisfaction2
6.12	happiness2
6.13	job_satisfaction2
6.14	afterlife2
6.15	job_satisfaction2
6.16	cholesterol
6.17	accident_seatbelt1
6.19	job_satisfaction3
6.21	happiness3
7.1	afterlife1
7.2	afterlife1
7.3	white_black_acceptance
7.4	AIDS_treatment
7.5	deathpenalty1
7.6	MBtest3
7.7	MBtest3
7.8	MBtest3
7.9	UCBAdmissions
7.1	accident_seatbelt3
7.12	accident_seatbelt2
7.13	government_spending

7.14	premarital_sex1
7.15	marijuana
7.16	accident_seatbelt2
7.21	government_spending
7.22	marijuana
7.24	birth_control
8.2	social_survey
8.8	Migraine_treatment
8.13	religious_belief_change
8.14	residence
8.15	sexual_behavior
8.16	environment_pro
8.17	environment_crisis
8.2	mutiple_sclerosis
8.23	Jour_cite
8.24	tennis