11 Statistical models in R

이번 세션은 당신이 회귀 분석이나 분산 분석 같은 통계적인 방법론을 어느 정도 알고 있다고 가정하고 쓴 것입니다. 후반부에서는 좀 더 욕심을 내서, 여러분이 Generalized Linear Model 과 비선형(Nonlinear) 회귀에 대해서도 알고 있을거라 가정하겠습니다.

통계 모형에 적합시키기 위한 여러 조건들은 상당히 잘 정의되어져 있어서 광범위한 문제들에 적용 가능한 일반적인 모형을 개발하는 것이 가능할 정도입니다.

R 은 통계 모형에 적합시는 것을 매우 단순하게 만들어주는 상호 작용 가능한 기능들의 조합을 제공합니다. 우리가 서문에서 살펴 본 것처럼, 기본적인 output 은 최소화 되어져 있고, 좀 더 상세한 내용이 필요하다면 이를 불러올 수 있는 출력 함수를 사용하면 됩니다.

11.1 Defining statistical models; formulae

통계 모형의 대표적인 예인 선형 회귀(Linear Regression) 모형은 독립적(Independent), 동질적이고(homoscedastic) 오차를 가집니다.

$$y_i = \sum_{j=0}^{p} \beta_j x_{ij} + e_i, \quad i = 1, \ldots, n,$$

이 모형에서 오차를 나타내는 e_i 는 독립적이고 동질적으로(IID, Independent and Identical) 평균이 0 이고 분산이 sigma^2 인 정규분포(N(0, sigma^2))를 따릅니다. 그리고 이것은 행렬(Matrix) 형태로는 이렇게 표현 됩니다.

$$y = X \beta + e$$

여기에서 y 는 반응변수 vector, X 는 model matrix 또는 design matrix 라 불리고, 각 설명 변수에 해당하는 x_0, x_1, ..., x_p 를 열로 갖는 행렬입니다. 많은 경우에 x_0 은 숫자 1 로만 구성된 하나의 열이며, 이것은 intercept term 으로 정의됩니다.

Examples

좀 더 수학적으로 정의를 구체화하기 전에, 도움이 될만한 예제를 몇 개 살펴보겠습니다.

y, x, x0, x1, x2, ...들이 숫자 변수들이라고 가정 합시다. X 는 하나의 행렬이고 A, B, C, ...라는 요인(Factor)들이 존재한다고도 가정합시다. 아래에서 왼쪽에 제시되는 공식(formulae)은 오른쪽에서 기술된 통계 모형을 구체화 합니다.

y ~ x
y ~ 1 + x
두 공식 모두 y 를 x 에 회귀시킨 단순 회귀모형을 의미합니다. 첫번째 공식에서는 intercept term 이 있다는 것이 함축되어져 있는 것이고, 두번째 공식에서는 intercept term 을 따로 표현해 준 차이가 있습니다.
y ~ 0 + x
y ~ -1 + x
y ~ x - 1
세 공식은 모두 원점을 지나는 y 를 x 에 회귀시킨 단순 회귀모형. (즉, intercept term) 이 없는 모형입니다.)

log(y) ~ x1 + x2
log 변환된 y, log(y)를 x1 과 x2 에 적합시킨 중(multiple)회귀 모형. (intercept term 은 함축되어져 있습니다.)

y ~ poly(x,2)
y ~ 1 + x + I(x^2)
모두 x 에 대한 2 차 다항(polynomial)회귀 모형. 첫번째 식에서는 직교(orthogonal) 다항을 사용하고 있으며, 두번째 식에서는 x 의 각 파워(power)들을 직접 basis 로 사용해서 표현했습니다.
*역주: 대부분의 통계 서적에서는 두번째 형태를 사용합니다.
y ~ X + poly(x,2)
행렬 X 와 x 의 2 차항을 함께 y 에 회귀시킨 중회귀 모형.
y ~ A
A 로 정의된 하나의 class 값에 대한 y 의 분산분석(ANOVA, analysis of variance)모형.
y ~ A + x
A 로 정의된 class 값들과 covariate X 를 사용한 y 의 covariate 모형에 대한 단순 classification 모형.
y ~ A*B
y ~ A + B + A:B
y ~ B %in% A

y ~ A/B

A 와 B 두개 요인의 non-additive 모형. 처음 두 방법은 모두 두 요인의 교차항을 사용하여

표현하였지만, 나중의 두 방법에서는 두 개의 classification 이 nested 되어 있음을 사용하여 표현한 것입니다. 이들을 좀 더 abstract 한 관점에서 보면, 4 가지 방법이 모두 같은 subspace 를 갖는 모형을 표현하고 있습니다..

y ~ (A + B + C)^2

y ~ A*B*C - A:B:C

모든 main effect 와 두 요인간의 interaction 만을 포함하는 삼요인 모형.

y ~ A * x

y ~ A/x

y ~ A/(1 + x) - 1

y 를 A 의 각 수준들로 이루어진 x 에 회귀시킨 단순 선형 모형을 세가지의 다른 coding 방법으로 나타낸 것. 가장 마지막 방법은 A 의 level 각각에 해당하는 서로 다른 intercept 와 slope 를 추정하는 모형을 의미합니다.

y ~ A*B + Error(C)

두 개의 treatments 에 해당하는 A 와 B 요인과 요인 C 에 의해 결정되는 error 부분을 포함하는 실험 모형. 예를 들면, 전체 실험 구성요소(plot)들 중 한 부분은 요인 C 에 의해 결정되고 있는 모형입니다.

R 에서는 연산자 ~ 이 하나의 *model formula* 를 표현하기 위해 사용됩니다.

일반적인 선형 모형의 경우, 다음과 같이 표현할 수 있습니다.

response ~ op_1 term_1 op_2 term_2 op_3 term_3 ...

이 경우

Response(반응 변수)

반응 변수(들)을 정의하는 하나의 벡터 혹은 행렬 (또는 이를 표현하는 식)

op_i

+ 또는 – 연산자, + 는 해당 변수가 그 모형에 포함되어 있음을 – 는 해당 변수가 그 모형에 빠져 있음을 의미합니다. (모형에 포함된 변수를 + 를 사용해서 따로 표현하는 것은 선택 사항 입니다.)

term_i

이것은 세 가지 방법으로 해석될 수 있는데

1) 하나의 벡터 또는 행렬, 혹은 1

2) 하나의 요인

3) 여러 개 요인들로 구성된 model formula. 이 경우 벡터 혹은 행렬 형태로 표현된 요인들이 연산자들에 의해 연결되어 표현됩니다..
어떤 경우에나 각각의 term 은 model matrix 에서 포함되거나 제외되는 열들을 모은것으로 정의할 수 있습니다.. 1 은 intercept 에 해당하는 열을 의미하는 것으로, 특별히 제외시켜야 하는 경우가 아니면, model matrix 에 이미 함축적으로 포함되어진 것입니다.

Formula 연산자들은 Glim 과 Genstat 가 만든 프로그램에서 사용된 Wilkinson 과 Rogers 의 표현 방법과 유사 합니다. R 에서의 유일한 큰 차이 점이라면 연산자 '.'이 formula 연산자에서는 ':'로 표현된다는 점일 겁니다. 이것은 '.'가 R 에서는 유효한 문자로 사용되기 때문입니다.

다음과 같은 방법으로 formula 연산자들이 사용됩니다. (based on Chambers & Hastie, 1992, p.29):

Y ~ M
: *Y* 는 M 에의해 모형 적합된다.

M_1 + M_2
: *M_1* 과 *M_2* *을 포함한다*.

M_1 - M_2
: *M_1* *은 포함하지만* *M_2* *는 제외한다*.

M_1 : *M_2*
: *M_1* 과 *M_2* *의* tensor product. 만약 두 term 이 모두 요인이라면, 이들의 subclass 들에 해당하는 요인..

M_1 %in% *M_2*
: 의미는 *M_1* : *M_2* 과 같음

M_1 * *M_2*
: *M_1* + *M_2* + *M_1*:*M_2*.

M_1 / *M_2*
: *M_1* + *M_2* %in% *M_1*.

M^n

n 차 interaction 들을 모함한 M 안에 든 모든 term 들을 의미

I(*M*)

복잡하게 표현된 M 을 따로 정의하는 방법. M 안에 든 모든 연산자들은 원래대로의 산술적 의미를 그대로 갖는다. 그리고 M 은 model matrix 에 사용된다.

대개 하나의 함수를 포함하는 괄호(()) 안에서는 모든 연산자들이 원래 그 연산자의 수학적 의미를 그대로 갖게 됩니다. 예를 들어 함수 I()의 경우 다른 여러 개의 산술 연산자로 표현 가능한 model formular 들을 포함할 수 있는 단일한 하나의 함수 개체인 것입니다.

특히 주의할 점은, model formula 들로 model matrix 의 열들을 표기할 때, 그것은 동시에 모형에서의 parameter 들을 표기하는 것이기도 합니다. 이것은 nonlinear model 과 같이 다른 형태의 방법론에는 해당되지 않는 내용 입니다.

11.1.1 Contrasts

우리는 model formula 들을 model matrix, X 에서의 각 열들을 사용하여 어떤식으로 표현할지를 알 필요가 있습니다. 만약 우리가 연속형 변수들만을 가지고 있다면, 하나의 열만을 가진 X 를 사용한 모형과 같은 방법으로 이것을 쉽게 표현할 수 있습니다. (그리고 그 모형에 1 로만 구성된 intercept term 이 포함된 경우도 같습니다.)

그렇다면 k-level 요인 A 의 경우는 어떨까요? 정답은 요인 A 가 순서화 되어 있는지 그렇지 않은지에 따라 달라집니다. 순서화 되어있지 않은 요인에 대해서는 2 번째 부터 k 번째까지의 level 을 의미하는 k-1 개의 indicator 들에 대응되는 k-1 개의 열을 사용할 것입니다. (즉, 이러한 parameterization 은 각 수준의 반응변수 값들을 첫번째 수준에서의 반응변수 값과 비교한다는 의미를 함축합니다.) 순서화된 요인에 대해서는, k-1 개의 열들이 상수항을 제외한 1 부터 k 까지의 직교 다차항들에 해당합니다.

정답이 이미 상당히 복잡해지긴 했지만, 아직 결론에 도달한 것이 아닙니다. 첫째로 intercept term 이 하나의 요인 형태로 포함되어 있어, intercept term 이 빠진 모형의 경우가 있을 수 있습니다. 이 경우 k 개의 열이 사용되어 각각의 열이 모두 각각의 level 에 대응됩니다. 두번째로는, 모든 형태가 contrast 를 어떤 형태로 표현하는지에 따라 전체 모형이 완전히 바뀌게 된다는 점이다. R 에서는 초기값(default)으로 contrast 가 다음과 같이 표현됩니다.

```
options(contrasts = c("contr.treatment", "contr.poly"))
```

이 점을 언급하는 이유는 R 과 S 가 순서화되지 않은 요인에 대해 서로 다른 초기값을 사용하고 있기 때문입니다. S 에서는 Helmert contrasts 를 사용하고 있지요. 그래서 만약 당신이 S 를 사용해서 기술된 책이나 논문의 결과를 R 에서 얻기 위해서는 위의 문장을 다음과 같이 표현해야 합니다.

```
options(contrasts = c("contr.helmert", "contr.poly"))
```

이렇게 R 에서 treatment contrast 를 채택한 것은 의도적인 것으로, 초보자들에게는 이 방법이 해석하기 쉬울 것이라 생각되었기 때문입니다.

model 에서 함수 contrasts 와 C 를 사용할 수 있도록 하기 위해, 좀 더 설명이 필요할 것 입니다. 우리는 아직 interaction term 에 대한 부분을 고려하지 않았는데요. 각각의 interaction term 은 그에 대응하는 열들의 곱 형태로 나타내어 집니다.

세부 사항들은 복잡해 보이지만, R 에서 사용되는 model formula 들은 보통 통계 전문가라면 가정할 만한 모형들을 다룰 수 있도록 하면서도 가능한 단순하게 설계된 것 입니다. 예를 들어, interaction term 은 포함하지만 주효과(main effects)를 포함하지 않는 모형의 경우 굉장히 특이한 결과를 내는데, 이런 모형에 대해서는 전문가들만 관심을 가질 것 입니다.

11.2 Linear models

보통의 여러 개 항을 가진 모형을 적합하는 가장 기본적인 방법으로 lm()이 있다. 그리고 이 함수를 사용하기 위한 간단한 방법은 다음과 같다:

> *fitted.model* <- lm(*formula*, data = *data.frame*)

예를 들면,

```
> fm2 <- lm(y ~ x1 + x2, data = production)
```

위의 표현은 y 에 대한 x1 과 x2 의 중회귀 모형에 적합시키는 방법입니다. (이 모형은 함축적으로 intercept term 을 포함하고 있습니다.)

데이터 production 을 사용한 모형에서 parameter 를 정할 때 중요한 점은 (기술적으로는 덜 중요한 부분일 수도 있지만) 모형에 포함될 변수들은 모두 데이터 프레임인 production 에 포함되어 있어야 한다는 점 입니다. 이러한 원칙은 데이터 프레임

production 이 탐색 경로안에 포함되어 있거나 그렇지 않은 경우에도 모두 지켜져야 합니다.

11.3 Generic functions for extracting model information

모형에 대한 정보를 보여주기 위한 일반(generic) 함수의 사용

lm() 함수는 적합된 모형에 대한 결과를 보여 줍니다: 기술적으로는 "lm"에 과 관련된 모든 결과들을 죽 나열한 것이지요. 적합된 모형에 대한 정보는 lm 이 포함하는 여러 개의 개체(object)들을 사용하는 보다 일반(generic) 함수들을 사용해서 보여주거나, 추출하거나, 그래프로 표현 가능합니다. 이러한 작업을 하기 위한 함수들로는

```
   add1   deviance   formula     predict  step
alias   drop1     kappa       print    summary
anova   effects   labels      proj     vcov
coef    family    plot        residuals
```

가장 많이 사용되는 함수들로는 다음과 같은 것들도 있습니다.

anova(*object_1*, *object_2*)

새로운 모형과 현재의 모형을 비교해서 분산분석(ANOVA) 결과를 출력하는 방법

coef(*object*)

회귀 계수를 출력하는 방법 (행렬 형태)

Long form: coefficients(*object*).

deviance(*object*)

잔차 제곱의 합, 혹은 특정한 경우 잔차들을 weighted 해서 구한 값

formula(*object*)

모형 formula 를 추출하는 방법

plot(*object*)

잔차, 모형에 적합된 값들 그리고 몇몇 diagnostic 의 결과들을 보여주는 네 개의 그림을 출력하는 방법

predict(*object*, newdata=*data.frame*)

이 함수를 사용하기 위해서는 새로 사용되는 데이터 프레임이 기존의 데이터 프레임과 같은 이름을 같는 변수들을 반드시 포함하고 있어야 합니다. 새 데이터 프레임 data.frame 안

에 포함되어진 변수들을 사용해서 예측된 값들의 벡터나 행렬 값을 생성.

print(*object*)

개체를 간단한 형태로 출력하는 방법. 사실 많은 경우 다른 함수에 포함되어져 가장 많이 사용되고 있는 함수.

residuals(*object*)

잔차 (혹은 그의 행렬)를 출력하는 방법. 필요한 경우, weighted 된 잔차 역시 출력 가능

Short form: resid(*object*).

step(*object*)

일종의 변수들에 대한 계층을 만들고 이를 바탕으로 각 term 들을 첨가하거나 제거함으로써 적절한 모형을 선택하기 위해 사용하는 방법. 가장 작은 AIC(Akaike's An Information Criterion) 값을 갖는 모형을 stepwise 방법으로 찾아내어 출력해준다.

summary(*object*)

회귀 분석 결과를 간결한 형태로 출력하는 방법

vcov(*object*)

적합된 모형안에 포함된 주요 parameter 들에 대한 variance-covariance(분산-공분산) 행렬을 출력하는 방법

11.4 Analysis of variance and model comparison

모형을 적합시키는 데 사용하는 함수의 일종인 aov (formula, data=*data.frame*)는 lm() 함수가 사용하는 방법과 비슷한 방법을 사용하는 가장 단순한 방법 중 하나 입니다. 또한, 위의 11.3 Generic functions for extracting model information 에서 제시된 대부분의 generic 함수들을 사용하는 것이 가능합니다.

매우 중요한 장점의 하나로, aov() 함수에서는 복잡한 형태의 오차 계층을 사용하여 모형을 분석하는 것이 가능하다는 점을 언급하지 않을 수 없는데요. 이러한 오차의 계층을 가정하기 때문에 split plot experiment 또는 block 간의 정보를 추가로 사용하는 balanced incomplete block design 이라는 모형 등을 다룰 수 있습니다.

모형에 대한 formula 는 다음과 같습니다.

response ~ mean.formula + *Error*(*strata.formula*)

strata.formula 부분을 통해서 오차의 계층를 표현할 수 있고, 이를 통해 여러 계층으로 이루어진 experiment 를 표현할 수 있는 것입니다. 가장 단순한 경우는 strata.formula 가 하나의 요인으로 표현되는 것이며, 이 경우 experiment 는 두 개의

계층을 가지는 것으로, 그 계층은 그 요인의 수준들에 대한 수준 내(within)와 수준 간(between)비교에 해당합니다.

예를 들면, 모든 주요 변수 요인들을 사용하는 model formula 의 하나로 다음과 같은 모형을 생각해 볼 수 있습니다:

> fm <- aov(yield ~ v + n*p*k + Error(farms/blocks), data=farm.data)
이 경우, v 와 n*p*k 에 대한 평균을 이용하고 세개의 오차 계층, 즉 ""between farms", "within farms, between blocks"그리고"within blocks 을 이용하는 모형을 기술하는 방법입니다.

11.4.1 ANOVA tables

여러 개의 적합 모형에 대한 분산분석(ANOVA)표 (혹은 표의 집합) 이라는 것도 역시 가능합니다. 잔차 제곱의 합은 일련의 순서에 해당하는 term 들이 하나씩 모형에 포함됨에 따라 계속 감소하는 것으로 나타납니다. 그러므로 orthogonal experiment 에 대해서는 변수가 첨가되는 순서가 그다지 중요하지 않다고 볼 수 있습니다.

또, 복잡한 계층을 가진 실험에 대해서는 반응 변수 값을 먼저 오차의 계층들 중 하나에 project 하고 각 projection 에 대한 평균을 이용해 모형을 적합시는 것을, 순차적으로 이를 반복함으로써 분산분석이 가능해 집니다. 좀 더 자세한 세부 사항에 대해서는 Chambers & Hastie (1992)를 참고하시기 바랍니다.

이렇게 default 로 지정되어 있는 full ANOVA 를 사용하는 대신 좀 더 간단한 대안으로 함수 anova()를 사용해서 두 세개의 관심 모형을 직접 비교하는 방법을 생각해 볼 수 있습니다.

> anova(*fitted.model.1*, *fitted.model.2*, ...)

이 경우 ANOVA 표는 모형들이 포함된 순서대로 서로 간에 얼마나 차이가 나는지를 보여주게 됩니다. 물론, 많은 경우 비교에 사용된 적합 모형들은 나름대로 어떤 계층적 순서를 갖도록 선택합니다. 이러한 모형의 순서는 default 에서와 다른 정보를 얻기 위한 것이 아니라, 오히려 모형을 이해하고 이들을 선택하는 것을 쉽게 하기 위해서 입니다.

11.5 Updating fitted models

update()함수는 이전에 정의되었던 적합 모형에서 단 몇 개의 term 을 첨가하거나 제거해서 새로운 모형으로 적합시킬 때 매우 유용한 함수입니다. 사용법은 다음과 같습니다.

> *new.model* <- update(*old.model*, *new.formula*)

new.formula 라는 특정 이름을 가진 새 함수를 정의함에 있어 update 를 사용하는 것은 " 기존의 old.model 에 새로운 모형이 대응된다"는 것을 의미하는 것입니다. 앞서 언급했던 대로, 함수의 이름으로 '.'가 포함될 수 있습니다. 좀 더 구체적인 예를 살펴보면,

```
> fm05 <- lm(y ~ x1 + x2 + x3 + x4 + x5, data = production)
> fm6  <- update(fm05, . ~ . + x6)
> smf6 <- update(fm6, sqrt(.) ~ .)
```

fm05 는 데이터 프레임 production 안에 포함된(혹은 포함되어 있을 것이라 추측되는) 다섯 개의 변수 x1 ~ x5 에 적합된 중회귀 모형 입니다. 여기에 여섯번 째 회귀 계수를 첨가해서 새로운 모형 fm6 을 적합했습니다. 그리고 여기에서 반응변수를 square root 로 변환시켜 모형을 적합시킨 것이 smf6 입니다.

특히, "data= "구문이 처음 모형 적합을 위한 문장에서 사용되면, 이에 해당하는 정보는 이후에 update()에 의해 생성되는 일련의 모형 적합에 계속해서 사용됩니다.

부호 '.'는 다른 용도로도 사용될 수 있지만, 이 경우 그 의미는 약간 달라집니다. 예를 들면, 다음과 같은 경우가 있습니다.

> fmfull <- lm(y ~ . , data = production)

이 모형은 y 를 반응 변수로 그리고 그 외 데이터 프레임 production 안에 들어 있는 다른 모든 변수들을 회귀계수로 사용하는 적합을 의미 합니다.

순차적으로 여러 개의 모형을 살펴보기 위한 함수로는 add1(), drop1() 그리고 step() 이 있습니다. 이러한 함수들은 이름의 의미 그대로 사용됩니다만, 좀 더 자세한 사용법은 온라인 help 를 참조하기 바랍니다.

11.6 Generalized linear models

Generalized linear model 은 정규분포를 따르지 않는 반응변수와 모형의 선형성(linearity)를 위한 단순하고 분명한 변수변환을 모두 사용해서 선형 모형(linear model)을 찾는 방법입니다. Generalized linear model 은 다음에 제시된 일련의 가정들에 의해 성립됩니다:

- 모형에서는 y 라는 반응 변수와 반응 변수의 분포에 영향을 주는 x_1, x_2, ..., 같은 stimulus 변수들이 중요합니다.

- Stimulus 변수들은 오직 하나의 선형 함수를 통해서만 y 의 분포에 영향을 미치게 됩니다. 이 선형 함수는 linear predictor 이며 다음과 같이 표기 됩니다.

  eta = beta_1 x_1 + beta_2 x_2 + ... + beta_p x_p,

따라서 beta_i 가 0 인 경우, x_i 가 y 의 분포에 미치는 영향력은 없다고 볼 수 있습니다. (if and only if)

- y 의 분포는 다음과 같은 형태 입니다.

f_Y(y; mu, phi) = exp((A/phi) * (y lambda(mu) - gamma(lambda(mu))) + tau(y, phi))

여기서 phi 는 (추정 가능한) *scale parameter* 이고, 모든 관측치는 상수(constant)이며, A 는 사전 가중치(prior weight)를 나타내며 이 가중치는 이미 주어졌다고 가정하지만 관측치에 따라 다른 값을 가질 수 있습니다. 또한, mu 는 y 의 평균입니다. 따라서, y 의 분포는 y 자신의 평균과 어떤 scale parameter 에 의해서 결정되는 것 입니다.

- 평균, mu 은 linear predictor(선형으로 연결된 설명 변수)에 대한 smooth invertible 함수로 표현될 수 있습니다: mu = m(eta), eta = m^{-1}(mu) = ell(mu)

여기서 역(inverse)함수인 ell()은 link function 입니다.

이러한 가정들은 실전에서 유용한 다양한 종류의 모형을 모두 포함할 수 있을 만큼 충분히 느슨한 것이지만, 동시에 근사적으로나마 추정(estimation)과 추론(inference)을 하기 위한 하나의 통일된 방법을 개발할 수 있을 정도 만큼 엄격한 것이기도 합니다. 이 방법에 관해서는 McCullagh & Nelder (1989) 이나 Dobson (1990) 과 같은 참고 서적을 통해 좀 더 자세하고 최신의 정보를 얻을 수 있습니다.

11.6.1 Families

R 에서 사용할 수 있는 generalized linear model 의 범주에는 반응 변수가 *gaussian*, *binomial*, *poisson*, *inverse gaussian* 와 *gamma* 분포를 따르는 경우를 포함하며, 또한 *quasi-likelihood* model 을 통해 반응 변수가 정확하게 이러한 분포를 따르지 않는 경우까지 포함 됩니다. 후자의 경우, *variance function* 이 평균에 대한 함수로 표현될 수 있어야만 하며, 다른 일반적인 경우에는 이 *variance function* 은 반응 변수의 분포에 의해서 결정됩니다.

반응 변수의 각 분포에서는 linear predictor 를 평균과 연결시키기 위한 link function 으로 여러 종류의 함수를 사용할 수 있습니다. 각각의 분포의 경우, 사용 가능한 link function 의 종류는 다음과 같습니다:

Family name	Link functions
binomial	logit, probit, log, cloglog
gaussian	identity, log, inverse
Gamma	identity, inverse, log
inverse.gaussian	1/mu^2, identity, inverse, log
poisson	identity, log, sqrt
Quasi-poisson	logit, probit, cloglog, identity, inverse, log, 1/mu^2, sqrt

반응 변수의 분포와 link function 의 선택 그리고 모형을 실행하기 위한 이와 관계된 다른 정보들의 조합은 generalized linear model 에서의 family 에 해당합니다.

11.6.2 The glm() function

반응 변수의 분포가 오직 stimulus 변수들에 대한 하나의 함수에 의해서만 영향을 받기 때문에, 이와 같은 메커니즘에 의해 generalized model 의 선형 부분이 표기될 수 있습니다. 이 경우, family 는 다른 방법으로 표현되어야 합니다.

generalized linear model 에 적합시키기 위한 R 함수는 glm()이고 다음과 같은 형태로 사용 됩니다.

> *fitted.model* <- glm(*formula*, family=*family.generator*, data=*data.frame*)

여기서 유일하게 새로 등장한 부분은 *family.generator* 이고, 이 부분이 family 를 기술하기 위한 도구 입니다. 실제로 모형을 정의하고 추정 수행하기 위한 일련의 함수와 표현들은 이 부분의 함수 이름에 의해 모두 결정되는 셈 입니다. 처음에는 약간 복잡하게 보일 수도 있지만, 굉장히 간단한 방법이지요.

family generator 를 설정하기 위해 표준적으로 사용되는 이름들의 목록은 이전 Families 부분에 포함된 표의 “Family Name” 항목을 통해 확인하실 수 있습니다. Link 의 선택도, family name 처럼 함수에 포함된 괄호 안에서 하나의 parameter 로 선택 가능합니다. 특히, quasi family 의 경우, variance function 역시 이런 식으로 표기 됩니다.

아래의 예제들이 이 방법을 이해하는데 도움이 될 것 입니다.

The gaussian family

gaussian family 를 사용하기 위한 방법은 다음과 같습니다.

```
> fm <- glm(y ~ x1 + x2, family = gaussian, data = sales)
```

다음 문장으로도 같은 결과를 얻을 수 있습니다.

```
> fm <- lm(y ~ x1+x2, data=sales)
```

하지만 이 경우는 위의 방법보다 훨신 효율이 떨어집니다. 또한, 이 방법은 여러 종류의 link 들 중 하나를 선택하는 함으로써 Gaussian family 가 된 것이 아니므로 실은 parameter 가 없는 모형 선택 방법인 것입니다. Nonstandard link 를 사용하는 Gaussian Family 를 사용해야 하는 경우라면, 이러한 문제는 quasi family 를 사용하여 해결할 수 있으며, 조금 위에 이에 대해 살펴보도록 하겠습니다.

The binomial family

Silvey (1970) 등장하는 간단한 예제를 하나 살펴보겠습니다. Kalythos 라는 한 에게해의 섬에서는 남성들이 선청성 안질환을 겪고 있습니다. 그 안질환은 나이가 들어감에 따라 좀 더 진행됩니다. 섬에 거주하는 다양한 연령대의 남성들이 실명(blindness)여부를 검사하기 위해 선택되었고 그 실험결과는 다음과 같이 기록되었습니다:

Age: 20 35 45 55 70

No. tested: 50 50 50 50 50

No. blind: 6 17 26 37 44

우리는 이 데이터에 대해 logistic 과 probit 두 가지 모형을 적합시키고 각 모형에서의 LD50 를 추정하는 문제에 관심이 있습니다. LD50 추정은 남성 거주자의 실명 확률이 50%가 되는 나이를 찾는 것입니다.

y 는 연령 x 에서의 실명한 사람의 수이고 n 이 검사를 한 사람의 수라면, probit 과 logit 모형 모두 다 y ~ B(n, F(beta_0 + beta_1 x))의 형태를 갖습니다. 단, probit 의 경우 F(z) = Phi(z)는 표준 정규 분포 함수이고, logit 의 경우 (default 로 사용되는 모형)에서는 F(z) = e^z/(1+e^z)입니다. 두 경우 모두, LD50 는 LD50 = - beta_0/beta_1 이며, 이것은 분포 함수 F(z)의 값을 0 으로 만들어주는 z 값에 해당합니다.

첫번째 단계는 모형에서 다루게 될 data 를 data frame 으로 만드는 것입니다.

```
> kalythos <- data.frame(x = c(20,35,45,55,70), n = rep(50,5),
                         y = c(6,17,26,37,44))
```

glm()를 사용해서 binomial model 을 적합 시키기 위해서는 반응 변수가 다음 세가지 중의 하나에 해당될 것 입니다:

- 반응변수가 벡터인 경우, 이 family 의 경우 데이터가 binary 데이터라는 점을 이미 가정하고 있으므로 이 벡터는 0 과 1 의 값 만을 포함합니다.

- 만약 반응변수가 2 열의 행렬 형태로 되어 있다면, 첫번째 열은 각 시행에서 성공한 숫자를, 두번째 열은 실패한 숫자를 의미 합니다.

- 만약 반응변수를 하나의 요인(factor)으로 되어 있다면, 첫번째 수준은 실패(0)로 그 외 나머지 모든 수준은 성공(1)로 간주해야 합니다.

이번 예제에서는 이러한 관습적 데이터 형태 중에서 두 번째 형태를 사용할 것이므로, 원래의 data frame 에 다음과 같은 행렬을 첨가해야 합니다:

```
> kalythos$Ymat <- cbind(kalythos$y, kalythos$n - kalythos$y)
```

모형을 적합시키는 방법은 다음과 같습니다.

```
    > fmp <- glm(Ymat ~ x, family = binomial(link=probit), data =
kalythos)
   > fml <- glm(Ymat ~ x, family = binomial, data = kalythos)
```

Logit link 는 default 이므로 두 번째 분장에서는 link 를 지정을 생략할 수 있습니다. 각 경우에 해당하는 적합 결과를 보기 위해 다음 문장을 사용할 수 있습니다.

```
       > summary(fmp)
   > summary(fml)
```

두 모형다 적합 결과가 좋습니다. (실제로 이 정도로 좋긴 힘듭니다.)

LD50 를 추정하기 위한 함수는 다음과 같이 같이 간단하게 정의 됩니다:

```
       > ld50 <- function(b) -b[1]/b[2]
   > ldp <- ld50(coef(fmp)); ldl <- ld50(coef(fml)); c(ldp, ldl)
```

이 데이터를 사용해서 직접 구한 LD50 의 추정치는 각각 43.663 과 43.601 입니다.

Poisson models

Poisson family의 경우 default link는 log이고, 실제로 이 family는 주로 빈도(frequency) 데이터를 Poisson log-linear로 적합시키는데 많이 사용됩니다. 원래, 이러한 빈도 데이터의 실제 분포는 많은 경우 multinomial 분포를 따르는데도 말이지요. 이 문제는 굉장히 방대하고도 중요한 주제 중 하나이므로 여기서는 다루지 않도록 하겠습니다. 이 모형을 사용하는 것은 non-gaussian generalized model의 사용의 거의 대부분을 차지할 정도로 중요합니다.

적지 않은 경우, 실전에서는 Possion data를 log나 square-root 변환시킨 후 Gaussian data처럼 분석하기도 했었습니다. 하지만, Poisson generalized linear이 후에 이러한 방법을 대체하게 되었고, 아래와 같은 방법으로 사용 가능 합니다:

```
> fmod <- glm(y ~ A + B + x, family = poisson(link=sqrt),data =
worm.counts)
```

Quasi-likelihood models

모든 family들의 대해, 반응변수의 variance는 mean에 의해 결정되고 scale parameter는 multiplier의 형태라는 공통점을 발견할 수 있습니다. Variance의 mean에 의한 영향력의 형태는 반응변수의 분포의 특성이기도 합니다: 예를 들면 반응변수가 Poisson 분포인 경우, Var(y) = mu.

quasi-likelihood를 이용한 추정과 추론은 정확한 반응변수의 분포를 찾아내는 것이 아니라, 오히려 link function과 variance function이 어떤 식으로 mean과 관련이 있는지와 관련이 있습니다. Quasi-likelihood를 이용한 추정은 gaussian distribution을 사용할 경우와 같은 방법들을 사용하기 때문에, 이 family는 사실 non-standard link function 또는 이와 동일하게 variance function을 사용한 Gaussian 모형 적합을 가능하게 해주었습니다.

예를 들어, 비선형 휘귀인 y = theta_1 z_1 / (z_2 - theta_2) + e에 대한 적합을 고려한다고 합시다. 이 모형은 y = 1 / (beta_1 x_1 + beta_2 x_2) + e의 형태로도 표현하는 것이 가능하며, 이때 x_1 = z_2/z_1, x_2 = -1/z_1, beta_1 = 1/theta_1, 그리고 beta_2 = theta_2/theta_1 입니다. 모형에 대응하는 적당한 데이터 프레임이 존재한다고 가정하면, 이 비선형 회귀는 다음과 같이 적합될 수 있습니다.

```
> nlfit <- glm(y ~ x1 + x2 - 1,
```

```
        family = quasi(link=inverse, variance=constant),
        data = biochem)
```

좀 더 자세한 설명이 필요하다면, 매뉴얼과 help 를 참조하시기 바랍니다.

11.7 Nonlinear least squares and maximum likelihood models

비선형 모형 중 어떤 특별한 형태에 해당하는 경우, Generalized Linear Models (glm())을 사용할 수 있습니다. 그러나 이 경우에도 대부분 비선형 최적화 방법 중의 하나인 비선형 곡선 적합이라는 관점으로 문제에 접근하고 있는 것입니다. R 에서는 비선형 최적화 방법을 위해 optim(), nlm()(R 2.2.0 부터 사용가능) 과 S-Plus 에서 ms() 과 nlminb()이 제공하던 것과 같은 (혹은 더 발전된) 기능들을 제공하는 nlminb()이 사용되고 있습니다. 이 방법은 몇몇 적합결여(lack-of-fit) 지표(index)들을 이용하여 이들을 최소화하는 parameter 값을 찾는 것 인데, 이러한 R 함수들은 여러 개의 parameter 값들에 대해 이 방법을 반복적으로 적용해서 모형을 적합시킵니다. 하지만, 선형 회귀 방법과는 다르게 이러한 procedure 가 만족스러운 수준의 추정치들로 converge 할 것이라는 점은 보장할 수 없습니다. 그래서 여기에 사용되는 모든 방법들을 사용할 때는 어떤 parameter 들을 대상으로 최적화를 할 것인지 그리고 covergence 가 초기값의 선택에 따라 크게 달라질 수도 있다는 점을 미리 고려해야 합니다.

11.7.1 Least squares

비선형 모형에 적합시키는 한 가지 방법은 제곱한 오차(error) 또는 잔차(residual)들의 합을 최소화시키는 것 입니다. 이러한 방법은 관측된 오차들이 정규 분포와 상당히 유사할 때 사용해야 합니다.

Bates & Watts (1988), page 51 에 나오는 예제를 하나 살펴 보겠습니다. 데이터가 아래와 같이 주어지고,

```
> x <- c(0.02, 0.02, 0.06, 0.06, 0.11, 0.11, 0.22, 0.22, 0.56, 0.56,1.10, 1.10)
    > y <- c(76, 47, 97, 107, 123, 139, 159, 152, 191, 201, 207, 200)
```

잔차 제곱합을 최소화하는 모형 적합 기준을 사용하기 위해 다음과 같은 함수를 정의 합니다.

```
    > fn <- function(p) sum((y - (p[1] * x)/(p[2] + x))^2)
```

모형 적합을 위해서는 parameter 들에 대한 초기 추정값이 필요합니다. 적절한 초기 값을 찾는 방법 중 하나는 데이터를 plot 하여, 몇 개의 parameter 값들을 추측해낸 후

이러한 추정값들을 사용했을 때의 모형 curve 를 원래 data plot 과 겹쳐 놓고 비교하는 것 입니다.

```
> plot(x, y)
    > xfit <- seq(.02, 1.1, .05)
    > yfit <- 200 * xfit/(0.1 + xfit)
    > lines(spline(xfit, yfit))
```

물론 더 나은 값이 존재할 수도 있겠지만, 일단 200 과 0.1 을 초기값으로 사용하는 것도 나쁘지 않을 것 같습니다. 다음으로 모형을 적합시키면:

```
> out <- nlm(fn, p = c(200, 0.1), hessian = TRUE)
```

모형 적합 후, out$minimum 은 SSE 를, out$estimate 은 parameter 들의 최소제곱(least squares) 추정치를 출력합니다. 추정치의 근사 오차(SE, standard errors)를 구하기 위한 방법은 다음과 같습니다:

```
> sqrt(diag(2*out$minimum/(length(y) - 2) * solve(out$hessian)))
```

위 명령분에 나타난 숫자 2 는 parameter 의 수를 의미 합니다. 95% 신뢰구간은 parameter 추정치에 1.96 SE 를 더하고 빼서 구합니다. 최소제곱 추정으로 구한 모형을 새로운 plot 으로 나타내 보겠습니다:

```
    > plot(x, y)
    > xfit <- seq(.02, 1.1, .05)
    > yfit <- 212.68384222 * xfit/(0.06412146 + xfit)
    > lines(spline(xfit, yfit))
```

기본 팩키지인 stats 은 비선형 모형을 최소제곱 방법으로 추정하기 위한 보다 다양한 기능들을 제공합니다. 바로 위에서 사용한 모형 적합 방법은 Michaelis-Menten 방법에 의한 것으로, 다음과 같은 방법으로 사용 가능합니다.

```
    > df <- data.frame(x=x, y=y)
    > fit <- nls(y ~ SSmicmen(x, Vm, K), df)
    > fit
    Nonlinear regression model
      model:  y ~ SSmicmen(x, Vm, K)
       data:  df
            Vm           K
```

```
212.68370711   0.06412123
 residual sum-of-squares:  1195.449
> summary(fit)

Formula: y ~ SSmicmen(x, Vm, K)

Parameters:
   Estimate Std. Error t value Pr(>|t|)
Vm 2.127e+02  6.947e+00  30.615 3.24e-11
K  6.412e-02  8.281e-03   7.743 1.57e-05

Residual standard error: 10.93 on 10 degrees of freedom

Correlation of Parameter Estimates:
     Vm
K 0.7651
```

11.7.2 Maximum likelihood

최대 우도방법(Maximum likelihood)은 오차들이 정규분포를 따르지 않는 경우에도 사용 가능한 비선형 모형 적합 방법 중의 하나 입니다. 이 방법은 log likelihood 를 최대화하거나, 이와 동등하게 Negative(음의 부호를 첨가한) log likelihood 를 최소화 시켜주는 parameter 값들을 찾아내는 방법 입니다. Dobson (1990), pp. 108–111 에 나오는 예제를하나 살펴보도록 하겠습니다. 이 예제에서는 dose-response 데이터를 logistic 모형에 적합시키고 있습니다. 물론, 이 경우 glm() 함수를 사용하는 것도 가능합니다.
데이터는 다음과 같습니다:

```
> x <- c(1.6907, 1.7242, 1.7552, 1.7842, 1.8113,
          1.8369, 1.8610, 1.8839)
> y <- c( 6, 13, 18, 28, 52, 53, 61, 60)
> n <- c(59, 60, 62, 56, 63, 59, 62, 60)
```

Negative log-likelihood 값을 최소화하기 위해서는:

```
> fn <- function(p)
  sum( - (y*(p[1]+p[2]*x) - n*log(1+exp(p[1]+p[2]*x))
        + log(choose(n, y)) ))
```

그럴 듯한 초기값을 선택한 후 모형을 적합시키겠습니다:

> out <- nlm(fn, p = c(-50,20), hessian = TRUE)

모형 적합된 후, out$minimum 은 negative log-likelihood 에 값을 out$estimate 은 maximum likelihood(최대 우도) 방법을 사용한 parameter 의 추정치들을 출력하게 됩니다. 추정치에 대한 SE 의 근사값을 구하는 방법은 다음과 같습니다.

> sqrt(diag(solve(out$hessian)))

95% 신뢰구간은 parameter estimate 에 1.96*SE 를 더한 값과 뺀 값 입니다.

11.8 Some non-standard models

이번 장의 마지막 순서로 특별한 형태의 회귀 및 데이터 분석을 하기 위해 R 에서 어떤 기능을 추가적으로 제공하고 있는지 간단히 언급하도록 하겠습니다.

- Mixed models.

  팩키지 nlme 에 든 lme() 와 nlme() 함수를 사용해서 선형과 비선형 혼합 효과(mixed effect) 모형을 다룰 수 있다. Mixted effect 모형이란 선형과 비선형 회귀모형을 의미하는 것으로 coefficient 들 중의 일부가 random effect 에 해당된다. 이러한 함수들은 모형을 기술하기 위해 많은 수의 formula 들을 이용한다.

- Local approximating regressions.

  함수 loess()는 일부 계수에만 weight 를 주는 가중회귀(weighted regression) 모형을 사용하려 비모수(nonparametric) 회귀를 적합시킨다. 이런한 회귀 모형은 지저분한 형태의 데이터의 추세를 파악하거나 굉장히 큰 크기의 데이터 셋에 대한 간단한 이해를 위해 데이터 차원을 축소시키는 데 사용될 수 있다.

  함수 loess()는 표준 팩키지의 하나인 stats 에 포함되어 있으며, projection 을 위한 코드를 함께 사용하면 회귀 분석을 할 수도 있습니다.

- Robust regression.

  데이터 안에 포함되어 있는 몇 개의 이상치(extreme value)들로 인한 영향력을 줄여 안정적인 회귀 모형에 적합시키기 위한 함수들은 여러 개 존재한다. 많이 사용되는 팩키지인 MASS 에서는 함수 lqs()를 제공하는데 이 함수는 안정적인 모형적합을 위한 최고의 방법이라 할 수 있다. 조금 덜 안정적이긴 하지만 좀 더 효율적인 방법으로는 MASS 팩키지 안에 든 rlm 함수를 사용하는 것 등이 가능하다.

- Additive models.

  이 방법은 이미 정해져 있는 변수들을 사용하여 회귀 모형을 만드는 경우에 사용되는 것으로, 변수들에 smooth additive 함수들을 적용한다. 각각의 변수에 하나의 smooth additive 함수를 대응시켜 사용하는 것이 일반적이다. R 에서는 사용자 개발 팩키지인 acepack 에서의 avas()와 ace() 함수 그리고 mda 에서의 bruto 와 mars 가 이러한 기능을 제공하는 예라 할 수 있다. 이를 확장한 것으로 Generalized Additive Model 이라는 것이 있으며, 이 방법은 gam 과 mgcv 같은 사용자 개발 팩키지에 의해 구현 가능하다.

- Tree-based models.

  예측이나 모형 해석을 위해서는 global 선형 모형을 찾는 대신, tree 형태의 모형에서는 데이터를 이미 결정된 변수들의 임계점(critical point)들에서 데이터를 양갈래로 나누어 궁극적으로 여러 개의 그룹으로 구성된 모형을 찾아낸다. 이러한 최종 그룹들은 가능한 그룹 내에서는 서로 동질적이며, 그룹 간에는 서로 다른 성질을 갖는다. 이러한 모형은 종종 다른 통계 방법으로는 볼 수 없는 데이터에 대한 새로운 관점을 제시한다.

  이러한 tree 모형은 또한 보통의 선형 모형의 형태로 표현될 수 있습니다. 이러한 tree 모형을 적합시키는 함수는 tree() 입니다. 하지만 plot()이나 text() 처럼 좀 더 일반적인 작업에 사용되는 함들일이 tree 형태의 모형에 적합된 결과를 시각적으로 표현하기 위해 효과적으로 사용되기도 합니다.

  R 에서는 tree 모형을 rpart 나 tree 같은 사용자 개발 package 들을 이용하여 구현할 수 있습니다.