2장 결측값 대치와 데이터 변환

CONTENTS

2.1 서론

- 예측 모델링을 포함한 다변량 자료 분석에서 첫 번째 장애물은 결측값(missing values)에 대한 처리이다. 대부분의 통계 분석 방법은 결측값을 포함하는 케이스를 제외한 뒤 완전한 자료에 대해 분석을 진행한다. 결측 자료의 양이 데이터셋의 크기에 비해 매우 작은 경우에는 이 방법이 분석의 편의(bias)를 없애기 위한 최선의 전략이 될 수 있다
- 그러나 이 방법은 경우에 따라 표본의 수를 크게 줄일 수 있으며, 중요한 정보를 없애게 되어 예측 모형의 성능을 크게 떨어뜨리는 원인으로 작용할 수 있다. 관측된 정보를 최대한 활용하기 위해서는 결측값에 대한 대치(imputation)가 중요하다.
- 한편, 몇몇 모형에서는 표준화 등의 변환된 자료 형태를 요구한다. 또한 모형의 성능 개선과 분석 목적에 따라 예측변수에 대한 변환이 필요하다. 이 장에서는 결측값 대치와 데이터 변환을 위한 R의 여러 함수를 소개한다.

2.2 결측값 대치

- 먼저, 결측값의 발생 유형에는 다음의 세 종류가 있다.
  - **MCAR**(missing completely at random): 결측값이 완전히 랜덤하게 발생한 경우이다. 예를 들어, 응답자가 우연히 질문을 뛰어넘은 경우이다. 이 경우 관측된 자료는 완비 자료로부터의 확률 표본으로 간주될 수 있다.
  - **MAR**(missing at random): 실제의 문제에서 가장 빈번한 형태로, 결측값이 관측된 다른 변수값에 영향을 받는 경우이다. 예를 들어, 젊은 응답자에 비해 나이든 응답자의 결측이 더 많은 경우이다. 그러나 응답자의 그룹 내에서는 자료가 MCAR이다.
  - **MNAR**(missing not a random): 결측값이 랜덤하게 발생된 경우가 아닌 경우로 보다 심각하다. 이 경우에는 데이터의 수집 절차를 확인하고, 발생 원인을 이해할 필요가 있다. 예를 들어, 설문 조사에 참여한 대부분의 사람들이 특정 질문에 답하지 않은 경우, 왜 그들이 답하지 않았는지? 질문이 불명확하지 않은지? 등을 확인할 필요가 있다.

2.2 결측값 대치

- 이 절에서는 자료가 MCAR과 MAR의 가정을 만족한다는 전제하에서 결측값에 대한 대치 방법을 다루기로 한다. MNAR의 경우에는 대치가 충분하지 않다 그 이유는 결측값이 유용한 데이터와는 완전히 다르기 때문이다.
- 결측값의 대치 방법에는 단일 대치와 다중 대치가 있다. 단일 대치(single imputation)는 한 개의 값으로 결측값을 대치하는 방법으로 평균, 회귀, EM 등이 있다.
- 반면, 다중 대치(multiple imputation, 이하 MI)은 결측값에 대해 여러 번의 대치를 수행하는 방법으로, 서로 상이한 여러 개의 대치 자료를 생성한다(보통 5~10개). 다중 대치는 결측값의 불확실성을 고려하므로 단일 대치에 비해 선호되나, 단점으로는 각 대치 자료에 대해 통계 분석을 반복 수행한 뒤 이를 결합하여야 하는 것이다.

2.2 결측값 대치

- 결측값의 비율은 문제가 될 수 있다. 단일대치의 경우 보통 안전한 최대 임계값은 대형 데이터 셋에 대해 전체의 5% 정도이다. 결측의 비율이 큰 경우에는 단일대치보다 다중대치를 사용하는 것이 적절하며, 변수나 자료를 제외하는 것도 고려할 수 있다.
- 여기서는 실제 예제를 통해 여러 종류의 R 패키지를 이용한 결측값의 대치 방법을 소개한다. 소개할 R 패키지와 함수는 다음과 같다.
  - mice{mice}
  - {Hmisc}의 impute()와 aregImpute() 함수

2.2 결측값 대치

예제 1 iris 자료에서 임의로 결측값을 발생시킨 자료에 대해 대치를 수행한다.

- R의 prodNA{missForest} 함수는 자료에서 인위적으로 결측값을 발생시켜준다.

```
> # 분석 자료: iris 자료에서 결측값을 10% 발생시킴
> library(missForest)
> set.seed(100)
> md <- prodNA(iris, noNA = 0.1)
```

2.2 결측값 대치

```
> head(md)
 Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1          NA          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6           NA         0.2    <NA>
6          5.4         3.9           NA         0.4  setosa
```

2.2 결측값 대치

(a) {mice} 패키지

- 이 패키지의 mice() 함수는 다변량 자료의 결측값에 대해 다중 대치(multiple imputation)를 제공한다. mice는 "multiple imputation by chained equations"의 약어이다.

```
> library(mice)
> md.pattern(md)    # 결측값의 패턴을 보여
 Sepal.Length Petal.Width Species Petal.Length Sepal.Width
90            1           1       1            1           1  0
 7            0           1       1            1           1  1
14            1           1       1            1           0  1
10            1           1       1            0           1  1
 8            1           0       1            1           1  1
 7            1           1       0            1           1  1
 1            0           1       1            1           0  2
 1            0           1       1            0           1  2
                                  (…)
```

2.2 결측값 대치

```
1             1             1             1             0             0  2
3             1             0             1             1             0  2
1             1             0             1             0             1  2
1             1             1             0             1             0  2
4             1             1             0             0             1  2
1             1             0             0             1             1  2
1             1             1             0             0             0  3
              9            13            14            18            21 75
```

해 석

결측이 없는 관측값이 90개, Sepal.Length에만 결측이 발생한 관측치가 7개이고, Sepal.Length에만 결측값이 총 9개 발생되었음을 알려준다.

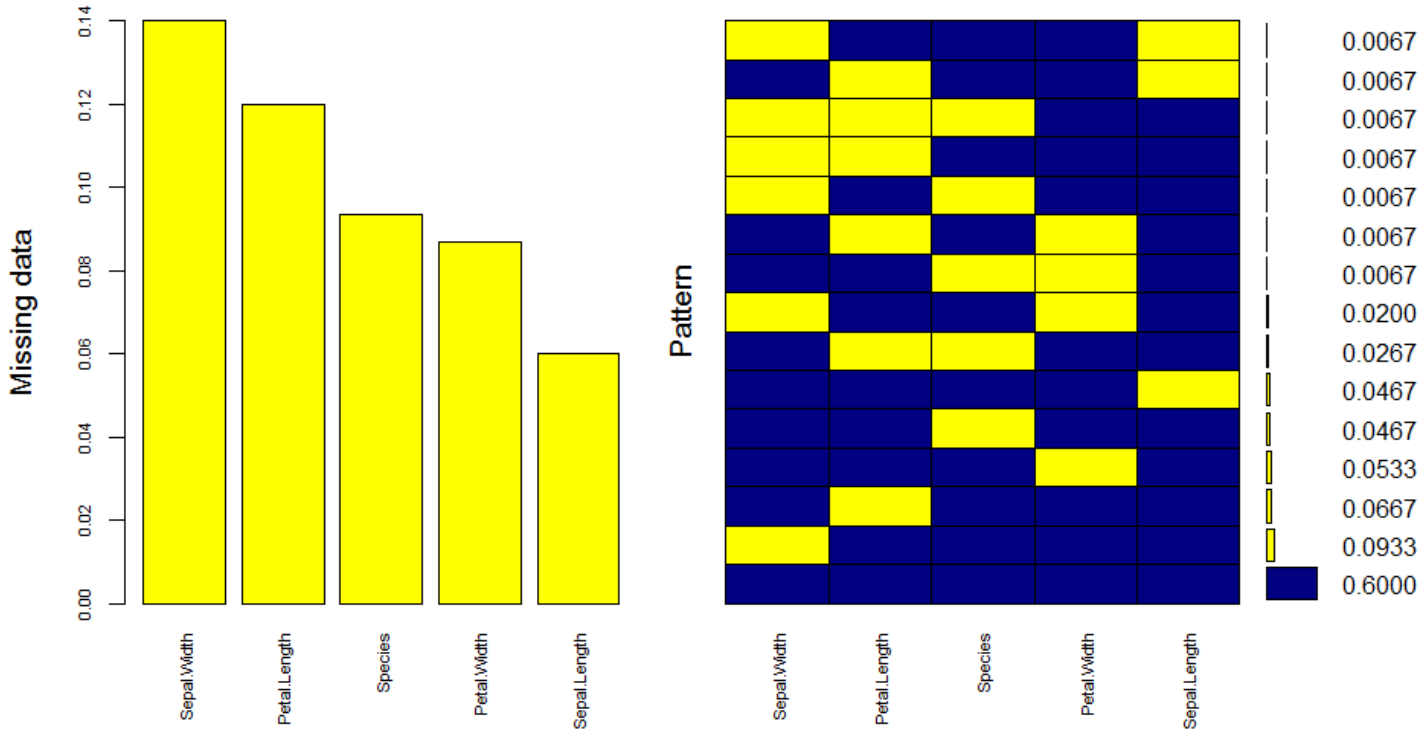
2.2 결측값 대치

참 고 aggr{VIM} 함수는 결측값의 패턴을 시각화하는 데 유용하다.

```
> library(VIM)
> plot.1 <- aggr(md, col=c('navyblue','yellow'),
                 numbers=TRUE, sortVars=TRUE,
                 labels=names(md), cex.axis=.7,
                 gap=3, ylab=c("Missing data","Pattern"))

 Variables sorted by number of missings:
     Variable       Count
  Sepal.Width 0.14000000
 Petal.Length 0.12000000
      Species 0.09333333
  Petal.Width 0.08666667
 Sepal.Length 0.06000000
```

2.2 결측값 대치


Missing data
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
Sepal.Width
Petal.Length
Species
Petal.Width
Sepal.Length
Pattern
Sepal.Width
Petal.Length
Species
Petal.Width
Sepal.Length
0.0067
0.0067
0.0067
0.0067
0.0067
0.0067
0.0067
0.0200
0.0267
0.0467
0.0467
0.0533
0.0667
0.0933
0.6000

2.2 결측값 대치

- R의 mice{mice} 함수를 이용하여 결측값을 대치한다. method= 옵션은 다음과 같다.
  - method= “pmm”, “logreg”, “polyreg”, “plor”
- method= 옵션을 생략할 경우, 변수의 유형에 따라 다음의 방법을 사용하여 대치를 수행한다.
  - pmm(predictive mean matching): 수치형 변수의 경우 디폴트
  - logreg(logistic regression): 이진(수준이 2개인) 변수에 대한 디폴트
  - polyreg(Bayesian polytomous regression): 요인(수준이 3개 이상인) 변수에 대한 디폴트
  - polr(proportional odds model): 2개 이상의 순서형 변수에 대한 디폴트

2.2 결측값 대치

- 아래에서 m=5는 다중 대치의 수를 5개로 지정하고, method = 'pmm'은 대치 방법으로 PMM 방법을 사용한다. PMM 방법은 결측값을 포함하는 각 관측치에 대해 해당 변수에 가장 가까운 예측 평균을 가진 (사용 가능한 값에서) 관측치를 찾는다. 이 "매치(match)"로부터 관측된 값을 대치값으로 사용한다.

```
> md.1 <- mice(md, m=5, maxit = 50, method = 'pmm', seed = 500)
```

- 결측값이 대치된 완비 자료는 complete() 함수를 이용하여 출력할 수 있다.

```
> # 결측값 대치 후의 완비 자료 출력(5개 중 2번째 사용)
> imputed.1 <- complete(md.1, 2)
```

2.2 결측값 대치

```
> # mice() 함수의 수행 결과 요약
> summary(md.1)
Multiply imputed data set
Call:
mice(data = md, m = 5, method = "pmm", maxit = 50, seed = 500)
Number of multiple imputations:  5
Missing cells per column:
Sepal.Length  Sepal.Width Petal.Length  Petal.Width      Species
           9           21           18           13           14
Imputation methods:
Sepal.Length  Sepal.Width Petal.Length  Petal.Width      Species
       "pmm"        "pmm"        "pmm"        "pmm"        "pmm"
VisitSequence:
Sepal.Length  Sepal.Width Petal.Length  Petal.Width      Species
           1            2            3            4            5
                                  (…)
```

2.2 결측값 대치

```
                                    (…)
PredictorMatrix:
             Sepal.Length Sepal.Width Petal.Length Petal.Width Species
Sepal.Length            0           1            1           1       1
Sepal.Width             1           0            1           1       1
Petal.Length            1           1            0           1       1
Petal.Width             1           1            1           0       1
Species                 1           1            1           1       0
Random generator seed value:  500
```

2.2 결측값 대치

(b) {Hmisc} 패키지

- 이 패키지의 impute() 함수와 aregImpute() 함수는 결측값 대치를 제공한다. impute() 함수는 사용자가 지정한 방법(평균, 중앙값, 최댓값, 최솟값 등)으로 결측값을 대치하며, aregImpute() 함수는 가법 회귀, 붓스트랩, PMM을 사용한 평균 대치를 수행한다. 특히. aregImpute() 함수는 변수의 형태를 자동으로 식별하고 이를 처리한다.

```
> library(Hmisc)
> # impute() 함수 이용: 평균값/임의값 대치
> # mean, min, max, median 지정 가능
> md.31 <- with(md, impute(Sepal.Length, mean))
> md.32 <- with(md, impute(Sepal.Length, 'random'))
```

2.2 결측값 대치

```
> # argImpute() 함수 이용
> md.33 <- aregImpute(~ Sepal.Length + Sepal.Width + Petal.Length
                                                  + Petal.Width + Species,
                                                  data = md, n.impute = 5)

> md.33
Multiple Imputation using Bootstrap and PMM

aregImpute(formula = ~Sepal.Length + Sepal.Width + Petal.Length +
    Petal.Width + Species, data = md, n.impute = 5)

n: 150  p: 5    Imputations: 5        nk: 3

Number of NAs:
 Sepal.Length  Sepal.Width Petal.Length  Petal.Width      Species
            9           21           18           13           14

                                  (…)
```

2.2 결측값 대치

```
                                    (…)
             type d.f.
Sepal.Length    s    2
Sepal.Width     s    2
Petal.Length    s    2
Petal.Width     s    2
Species         c    2

Transformation of Target Variables Forced to be Linear

R-squares for Predicting Non-Missing Values for Each Variable
Using Last Imputations of Predictors
Sepal.Length  Sepal.Width Petal.Length  Petal.Width      Species
       0.899        0.585        0.982        0.957        0.990
```

해 석 위 결과에서 R^2(R-squares)는 예측된 결측값에 대한 성능을 나타낸다. 이 값이 클수록 예측 성능이 우수하다.

2.3 데이터 변환

- 모형의 적합에 앞서 예측변수에 대한 변환을 소개한다. 자료의 변환에는 분석 모형과 분석 목적에 따라 분석자의 주관적인 판단이 요구된다. 이 절에서는 여러 가지 변환 방법과 함께 R을 이용한 변환 절차를 소개한다.

2.3 데이터 변환

2.3.1 박스-콕스 변환

- R 패키지 {caret}의 preProcess() 함수는 개별 예측변수에 대한 중심화(centering), 척도화(scaling) 및 박스-콕스 변환(Box-Cox transformation)을 제공한다.

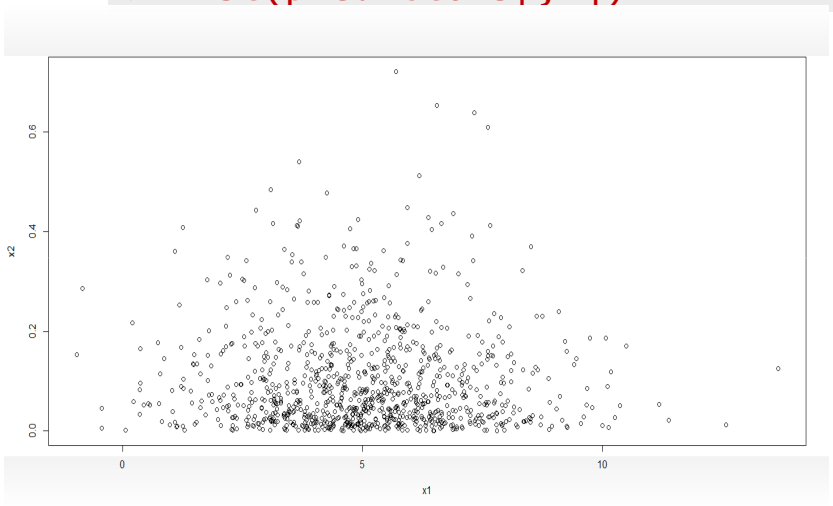
예제 2 두 개의 예측변수(x_1,x_2)를 가지는 데이터프레임을 생성하고, 이를 이용하여 전처리 과정을 수행한다.

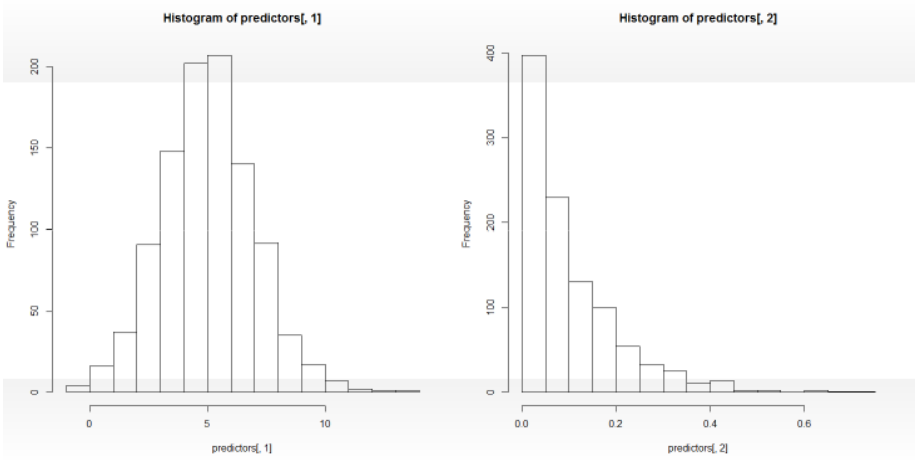
```
> set.seed(200)
> predictors = data.frame(x1=rnorm(1000, mean=5, sd=2),
                          x2=rexp(1000, rate=10))
```

2.3 데이터 변환

- 데이터프레임 predictors를 그림으로 나타내면 다음과 같다.

```
> plot(predictors)
> par(mfrow=c(1,2))
> hist(predictors[,1])
> hist(predictors[,2])
```





2.3 데이터 변환

- 일반적으로 Box-Cox 변환은, 양의 값을 가지는 자료에 적용되는, 분산안정화 및 정규화를 위한 변환으로 알려져 있다. Box-Cox 변환(또는 멱(power) 변환)은 다음과 같다.

$$x^* = \begin{cases} \dfrac{x^\lambda - 1}{\lambda}, & \lambda \neq 0 \\ \log(x), & \lambda = 0 \end{cases}$$

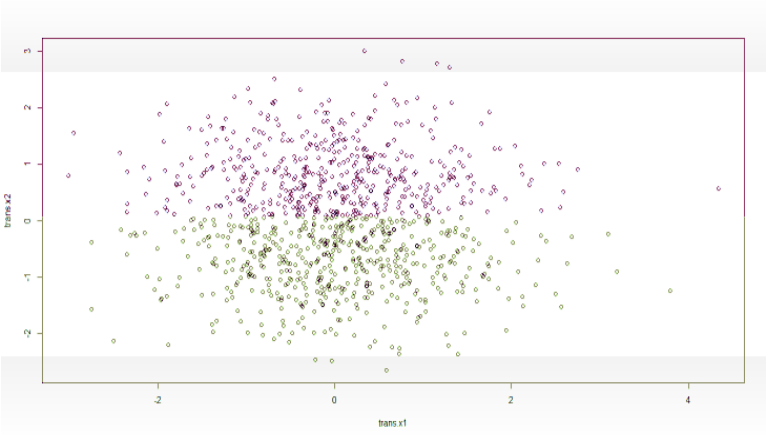
- 위의 자료에 대해 변환의 차수인 λ를 추정하고, 그 결과를 위의 자료에 적용한 뒤, 각 변수에 대해 중심화와 척도화를 함께 수행한다.

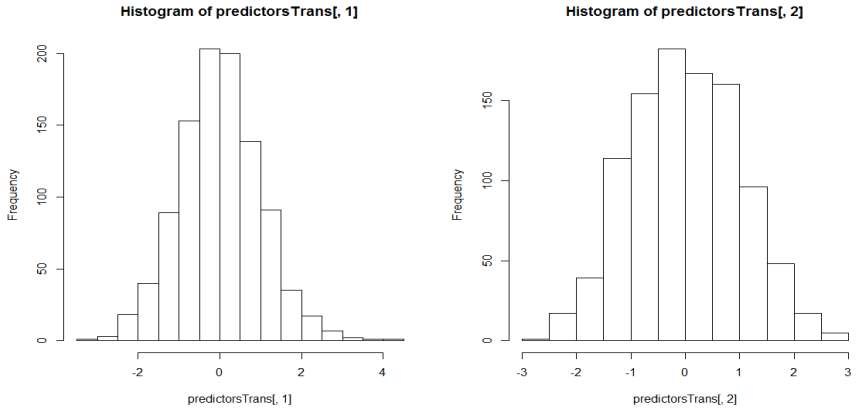
```
> require(caret)
> trans = preProcess(predictors, c("BoxCox", "center", "scale"))
> predictorsTrans = data.frame(trans = predict(trans, predictors))
```

2.3 데이터 변환

- preProcess() 함수를 통해 생성된 새로운 데이터프레임(predictorsTrans)을 그려보면 다음과 같다.

```
> plot(predictorsTrans)
> par(mfrow=c(1,2))
> hist(predictorsTrans[,1])
> hist(predictorsTrans[,2])
```





2.3 데이터 변환

참 고

1. R에서 박스-콕스 변환을 수행하는 함수에는 boxcox{MASS}, bcPower{car}, powerTransform{car} 등이 있다.

2. 또한, 자료의 변환을 포함한 R에서 제공하는 전처리 함수에는 scale{base}, ScaleAdv{pcaPP}, stdize{pls}, PreProcess{caret}, normalize{sparseLDA} 등이 있다. 이 가운데 처음 3개 함수는 중심화와 척도화만 제공한다. PreProcess{caret} 함수는 매우 다양한 예측변수에 대한 전처리 변환을 제공한다.

2.3 데이터 변환

참 고

분산안정화 변환

많은 통계적 자료들은 분산이 평균과 관계를 맺고 있다. 예를 들어, 다른 모집단의 비교에서 소득의 분산은 평균 소득에 따라 커지는 경향이 있다. 분산안정화 변환(variance stabilizing transformation, 이하 VST)의 목적은 분산과 평균의 관계성을 제거하는 것이다. 분산안정화를 위한 변환은 자료의 형태에 따라 달라지며 그 예는 다음과 같다.

- 비율(또는 이항) 자료: 역사인 또는 로짓 변환
- 개수(또는 포아송) 자료: 제곱근(또는 Anscombe) 변환
- 표본상관계수: 피셔의 변환
- 회귀분석에서의 박스-콕스 변환

VST의 이론에 대해서는 나종화(2016)의 '수리통계학'을 참고하기 바란다.

2.3 데이터 변환

2.3.4* 반응변수의 변환: 역반응그림

- 여기서는 역회귀(inverse regression)를 사용한 반응변수(Y)의 변환을 소개한다. 반응변수와 예측변수 간에 다음의 관계가 성립된다고 하자.

$$Y = g(\beta_0 + \beta_1 x + \epsilon).$$

- 우리의 목적은 다음의 관계가 만족되도록 반응변수의 변환 $g^{-1}()$를 찾는 것이다.

$$g^{-1}(Y) - \beta_0 + \beta_1 x + \epsilon.$$

- 이 변환은 정규화(nomality)를 위한 변환이 아니라, 선형화(linearity)를 위한 변환이다. 그러나 변환의 결과는 종종 박스-콕스의 변환과 유사한 결과를 제공한다.

$$g$$

2.3 데이터 변환

- 이 변환을 찾는 과정은 다음과 같다. 먼저, 다음의 선형회귀모형을 적합한다.

$$Y = \beta_0 + \beta_1 x + \epsilon.$$

- 다음으로 관측값(y)을 x축(수평축), 추정값($\hat{y}$)을 y축(수직축)으로 하는 그림을 그린다(통상적으로 y는 수직축에 그려지나, 이 그림에서는 수평축에 놓음으로 이 그림을 "역반응(inverse response) 그림"이라고 함). 이 그림이 직선적인 경우에는 반응변수에 대한 변환이 불필요할 것이다. 역반응 그림이 직선의 패턴을 따르지 않을 경우, 회귀의 결과 잔차제곱합(residual sum of squares, 이하 RSS)을 최소화하는 반응변수의 변환을 구할 수 있다.
- R의 invResPlot() 함수는 역반응 그림과 함께 RSS를 최소화하는 반응변수의 박스-콕스 변환 차수를 제공한다.

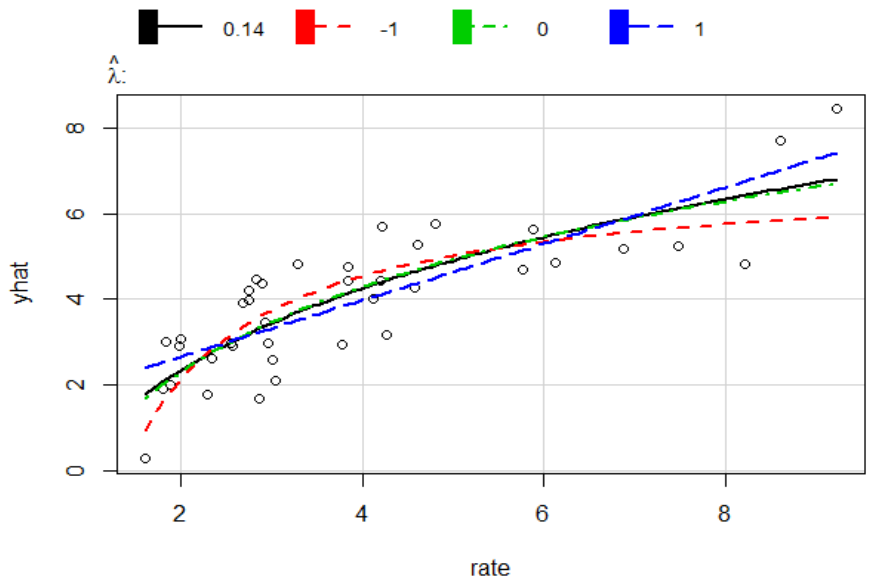
2.3 데이터 변환

예제 3 고속도로 사고에 관한 Highway1{car} 자료에 대해 역반응 그림을 통한 변환을 수행한다.

```
> library(car)
> m <- lm(rate ~ log(len) + log(adt) + slim + shld + log(sigs1),
          Highway1)
> invResPlot(m)
      lambda       RSS

1  0.1350783  31.57739
2 -1.0000000  35.45785
3  0.0000000  31.63514
4  1.0000000  33.68958
```

2.3 데이터 변환



해 석

RSS를 최소화하는 변환차수가 영(0)에 가까우므로 반응변수에 대해 로그 변환이 필요함을 알 수 있다. 역변환 그림의 장점 가운데 하나는 변환의 선택에 따라 개별 관측치들의 지레(leverage)와 영향(influential)을 시각화 해 준다는 것이다. 즉, 위 그림에서 변환선으로부터 크게 벗어난 점은 해당 변환에서 영향점으로 간주될 수 있다.

2.3 데이터 변환

- 대안적인 방법으로, 잔차의 정규화를 위한 박스-콕스 변환의 차수를 구하면 다음과 같다.
  그 결과, 역반응 그림에서와 마찬가지로, 반응변수에 대해 로그 변환이 적합함을 확인할 수 있다 ($\lambda \approx -0.253$).

```
> summary(powerTransform(m))
bcPower Transformation to Normality
   Est.Power Std.Err. Wald Lower Bound Wald  Upper Bound

Y1    -0.253   0.2891           -0.8197            0.3137
Likelihood ratio tests about transformation parameters
                               LRT df         pval
LR test, lambda = (0)  0.7749801  1 3.786808e-01
LR test, lambda = (1) 18.6129697  1 1.601274e-05
```