R によるロジスティックモデル

青木繁伸

2015 年 6 月 24 日

目次

1　直線回帰とロジスティック回帰の違い

図 1 のように，従属変数が 2 値変数（2 種類の値のどちらかしかとらない）場合に，直線回帰を行うのは不適切である。例えば，0/1 データの場合，[0, 1] の値をとるということは，1 の事象が起きる確率として解釈できるが，0 未満の値や 1 を超える値の解釈ができない。

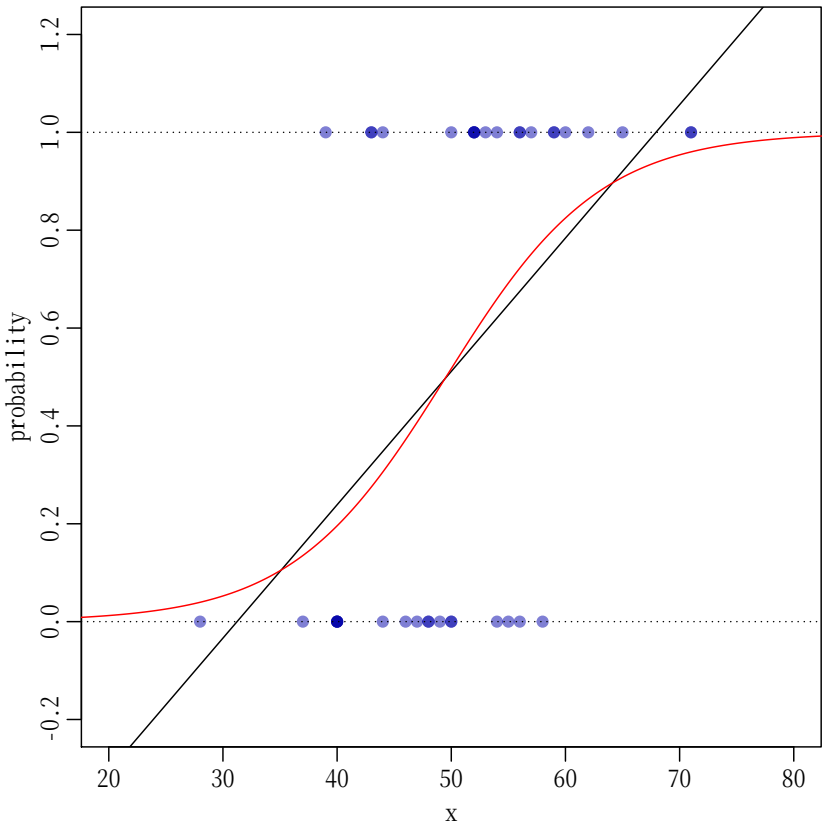

図 1　ロジスティック回帰分析と重回帰分析の違い

ある事象が発生する（従属変数が 1 になる）確率を P としたとき，$\dfrac{P}{1-P}$ はオッズ[*1]，その対数をとった $\log\left(\dfrac{P}{1-P}\right)$ はロジットまたは対数オッズと呼ばれる。

（1）式のようにロジットが独立変数の線形結合式 $\lambda = b_0 + b_1\, x_1 + b_2\, x_2 + \ldots + b_p\, x_p$ で表せるとするのがロジスティックモデルである。λ を線形予測子 linear predictor ともいう。

$$\log\left(\frac{P}{1-P}\right) = b_0 + b_1\, x_1 + b_2\, x_2 + \ldots + b_p\, x_p = \lambda \tag{1}$$

（1）式の両辺の逆対数をとった（2）は，従属変数が 1 になる確率と 0 になる確率の比をとったもの（オッズ）である。

$$\frac{P}{1-P} = \exp(b_0 + b_1\, x_1 + b_2\, x_2 + \ldots + b_p\, x_p) \tag{2}$$

x_1 以外の独立変数が同じであるという条件下で，$x_1 = u + 1$ のときのオッズと $x_1 = u$ のときのオッズの比をとると（3）式のようになる。これは，オッズ比と呼ばれ，x_i が 1 単位増えたときに，オッズは $\exp(b_i)$ 倍に

[*1] オッズというのはギャンブルで「見込み」を表すために使われるものである。$\dfrac{\text{成功の回数}}{\text{失敗の回数}}$ で計算されるもので，日本の公営競馬. 競輪などにおけるオッズとは定義が違う。分子と分母をそれぞれ「成功の回数と失敗の回数の合計」で割れば，$\dfrac{\text{成功の確率}}{\text{失敗の確率}} = \dfrac{\text{成功の確率}}{1 - \text{成功の確率}}$ となる。成功とは「注目している事象が起きること」とする。

なることを意味する。

$$\frac{P_{x_1=u+1}/(1-P_{x_1=u+1})}{P_{x_1=u}/(1-P_{x_1=u})} = \frac{\exp(b_0 + b_1\,(u+1) + b_2\,x_2 + \ldots + b_p\,x_p)}{\exp(b_0 + b_1\,u + b_2\,x_2 + \ldots + b_p\,x_p)} = \exp(b_1) \tag{3}$$

(1) 式を P について解くと (4) 式のようになる。

$$P = \frac{1}{1+\exp(-\lambda)} \tag{4}$$

P は [0, 1] の値をとり，λ を横軸，P を縦軸として描かれる曲線はロジスティック曲線と呼ばれる[*2]。

1.1 相対危険度とオッズ比

夜，交通事故にあった歩行者の服装の色を調べたところ，表 1 のようになっていたとする。

表 1　服装と交通事故

	結果		
原因	交通事故にあった	交通事故にあわなかった	合計
黒っぽい服装	240	7760	8000
明るい服装	20	1980	2000

この表では，黒っぽい服装で交通事故にあったのは 3%，明るい服装で交通事故にあったのは 1% であった。前者は後者より 2% 高いといえる。このような集計表を評価するときに，医学や保健学の分野などでは[*3]パーセントで表された数値に差があるかどうかを見るほかに，相対危険度とかオッズ比を使うことがある。

相対危険度というのは，原因のあるグループで結果が起きる確率が，原因のないグループで結果が起きる確率の何倍かを表すものである。黒っぽい服装をしていて交通事故にあった人の割合は 3%，明るい服装をしていたのに交通事故にあった人は 1% で，その比をとると 3 なので，黒っぽい服装をしている人は明るい服装をしている人に比べて交通事故にあう危険性は 3 倍だということになる。相対危険度が 1 より大きい場合は原因があることで結果がより起こりやすいこと，1 より小さい場合は逆に原因があると結果が起こりにくいことを表す。1 の近辺の値を取るときには原因の有無は結果の有無とはあまり関係がない（ちょうど 1 なら完全に無関係）ということになる。一般的には相対危険度が 3 以上（3 分の 1 以下）になれば原因と結果に強い関係があると判断されるようだ（喫煙と肺癌の相対危険度は 1 以上，場合によっては 10 以上という研究結果もある）。

相対危険度は危険性をわかりやすく評価できるというメリットがあるが，原因の有無ごとに計算される割合の精度を保証するためには分母がある程度大きくなければならないなどいくつかの条件が必要である。そこで，そのような条件を完全には見たさない場合にも相対危険度の推定値として使える指標として，オッズの比を取ったオッズ比というものがある。オッズというのはギャンブルで「見込み」を表すために使われるものである。「成功の回数/失敗の回数」で計算されるもので，日本の公営競馬. 競輪などにおけるオッズとは定義が違う。分子と分母をそれぞれ「成功の回数と失敗の回数の合計」で割れば，「成功の確率/失敗の確率＝成功の確率/（1 －成功の確率)」となる。成功とは「注目している事象が起きること」とする。今の場合だと「交通事故にあう」というのが「成功」である。黒っぽい服装の場合のオッズは 240/7760 ≒ 0.031，明るい服装の場合には 20/1980 ≒ 0.010 となる。原因のある場合のオッズを分子，原因のない場合のオッズを分母にとって，比を計算したものがオッズ比である。つまり，(240/7760)/(20/1980) = (240 × 1980)/(7760 × 20) ≒ 3.061856 となり，確かに相対危険度（3）に近い値になっている。

[*2] 同じような「S 字状曲線」を表すプロビットモデル（プロビット曲線）があるが，数学的に簡単に取り扱えるロジスティックモデルのほうがよく使われる。

[*3] 社会学や心理学の分野でも広く使われている。

ロジスティックモデルの種類

多重ロジスティックモデル　従属変数は 2 値データ

多項ロジスティックモデル　従属変数は 3 つ以上のカテゴリーを持つ名義尺度データ

順序ロジスティックモデル　従属変数は 3 つ以上のカテゴリーを持つ順序尺度データ（カテゴリーに順序関係がある）

累積ロジスティックモデル

比例オッズモデル

2　多重ロジスティックモデル

多重ロジスティックモデルは，従属変数が 2 値データである場合に適用される。

2.1　stats パッケージの glm 関数

表 2　二値データ

	No.	x1	x2	y	No.	x1	x2	y	No.	x1	x2	y
1	1	57	47	0	11	54	59	0	21	54	53	0
2	2	54	48	0	12	36	45	1	22	50	61	1
3	3	50	37	0	13	31	36	1	23	53	52	0
4	4	69	63	0	14	43	40	0	24	58	40	1
5	5	44	56	1	15	24	40	1	25	64	71	0
6	6	48	36	1	16	48	46	1	26	51	34	0
7	7	39	65	0	17	35	57	0	27	54	42	1
8	8	49	40	1	18	47	47	1	28	42	44	1
9	9	54	60	0	19	59	66	0	29	56	46	0
10	10	67	56	0	20	53	57	1	30	53	55	0

表 2 のようなデータに対して stats パッケージの glm() を適用する。結果は，summary() で表示する。

```
> ans.glm <- glm(y~x1+x2, data=d, family=binomial)
> summary(ans.glm)
Call:
glm(formula = y ~ x1 + x2, family = binomial, data = d)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6335  -0.8518  -0.3430   0.9606   1.7023

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  8.13524    3.58443   2.270   0.0232
x1          -0.09798    0.05707  -1.717   0.0860
x2          -0.07235    0.04794  -1.509   0.1313

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 41.054  on 29  degrees of freedom
Residual deviance: 32.227  on 27  degrees of freedom
AIC: 38.227

Number of Fisher Scoring iterations: 4
```

求められた係数は対数オッズ比なので，指数をとればオッズ比になる。

```
> coef(ans.glm)
(Intercept)          x1          x2
 8.13524408 -0.09797718 -0.07234616
> exp(coef(ans.glm))
 (Intercept)           x1           x2
3412.6489238    0.9066696    0.9302088
```

オッズ比の信頼区間は `confint()` で対数オッズの信頼区間を求めたものの指数をとることで得られる。

```
> exp(confint(ans.glm))
                2.5 %       97.5 %
(Intercept) 8.7941746 1.589734e+07
x1          0.7960658 1.002113e+00
x2          0.8387196 1.017469e+00
```

`glm()` が返すオブジェクト `linear.predictors`（線形予測子; LP としよう）は，元のデータを `coefficients` により線形変換した合成変数である。

`fitted.values` は線形予測子に基づいて $\dfrac{1}{1+\exp(-LP)}$ により計算される。

$$LP = 8.13524 - 0.09798\, x1 - 0.07235\, x2$$

```
> (coeff <- ans.glm$coefficients)
(Intercept)          x1          x2
 8.13524408 -0.09797718 -0.07234616
> LP <- coeff[1] + data.matrix(d[, 1:2]) %*% coeff[2:3]
> FV <- 1/(1+exp(-LP))
> head(cbind(LP, ans.glm$linear.predictors, FV, ans.glm$fitted.values))
        [,1]       [,2]      [,3]      [,4]
1 -0.8497248 -0.8497248 0.2994906 0.2994906
2 -0.6281394 -0.6281394 0.3479325 0.3479325
3  0.5595771  0.5595771 0.6363547 0.6363547
4 -3.1829896 -3.1829896 0.0398109 0.0398109
5 -0.2271370 -0.2271370 0.4434586 0.4434586
6  0.8278776  0.8278776 0.6959060 0.6959060
```

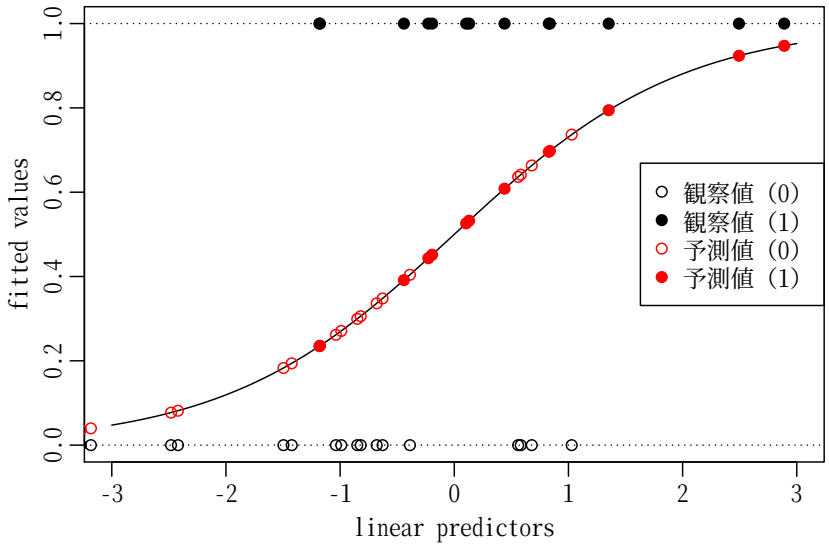

図 2　観察値と線形予測子と予測値

2.2 尤度比検定

尤度比検定は，複雑なモデルと単純なモデルそれぞれの尤度比の対数をとったものの −2 倍がそれぞれの自由度の差を自由度とする χ^2 分布に従うことを利用する検定法である。

具体的には，あるモデルが有意なものかどうかを独立変数を使わないモデル（ヌル・モデル）と比較するという場合を考えよう。

```
> logLik(ans.glm)
'log Lik.' -16.11352 (df=3)
> sum(ifelse(d$y, log(ans.glm$fitted.values), log(1-ans.glm$fitted.values)))
[1] -16.11352


> ans0.glm <- glm(y~1, data=d, family=binomial)
> summary(ans0.glm)
Call:
glm(formula = y ~ 1, family = binomial, data = d)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.066  -1.066  -1.066   1.293   1.293

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.2683     0.3684  -0.728    0.467

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 41.054  on 29  degrees of freedom
Residual deviance: 41.054  on 29  degrees of freedom
AIC: 43.054

Number of Fisher Scoring iterations: 4
```

前項で示した x1，x2 を独立変数として y を予測した場合には，デビアンス[*4]が 34.864，AIC[*5] が 40.864 であった。その分析例にも示されていたが，改めて独立変数を使わないモデルのデビアンスと AIC を求めると，それぞれ 41.054 と 43.054 であった。比較すると，当然ながら x1，x2 を独立変数としたモデルの方が優れていることはわかるが，有意に優れているかどうかはこの数値だけからはわからない。

そこで 1 つの方法として，anova() を使う方法を示す。glm() が返すオブジェクトを anova() に与えればよいのだが，anova() は P 値を計算しない。そのため，anova() が返すオブジェクトに含まれる Deviance と Df 要素を用いて，自分で pchisq() を使って計算しなければならない。

```
> (res.anova <- anova(ans0.glm, ans.glm))
Analysis of Deviance Table

Model 1: y ~ 1
Model 2: y ~ x1 + x2
  Resid. Df Resid. Dev Df Deviance
1        29     41.054
2        27     32.227  2   8.8269
> pchisq(res.anova$Deviance, res.anova$Df, lower.tail=FALSE)[2]
[1] 0.01211352
```

[*4] 逸脱度と訳されることがある。モデルからの逸脱の度合いをあらわすものである。logLik() で求められる対数尤度を用いて，デビアンス = −2 × 対数尤度

[*5] AIC= デビアンス + 2 × (パラメータ数 + 1)

あるいは，glm() はヌル・モデルについてのデビアンスと自由度も返すので，ヌル・モデルとの比較ならば，以下のようにしても良い。

```
> pchisq(ans.glm$null.deviance-ans.glm$deviance, ans.glm$df.null-ans.glm$df.residual,
+        lower.tail=FALSE)
[1] 0.01211352
```

これを関数にしておけばいつでも手軽に尤度比検定を行うことが出来る。

```
> lrt <- function(obj) {
+         lr <- obj$null.deviance-obj$deviance
+         df <- obj$df.null-obj$df.residual
+         p.value <- pchisq(lr, df, lower.tail=FALSE)
+         return(list(lr=lr, df=df, p.value=p.value))
+ }
> lrt(ans.glm)
$lr
[1] 8.826866

$df
[1] 2

$p.value
[1] 0.01211352
```

または，対数尤度の「−2 倍」がデビアンス[*6]という関係があるので，以下のように計算してもよい。

```
> L0 <- logLik(ans0.glm)
> L1 <- logLik(ans.glm)
> L01 <- as.vector(- 2 * (L0 - L1))
> df <- attr(L1, "df") - attr(L0, "df")
> pchisq(L01, df, lower.tail = FALSE)
[1] 0.01211352
```

独立変数を 1 つずつ加えていくというモデルの吟味を行う場合には，以下のようにする。

```
> anova(ans.glm, test="Chisq")
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                    29     41.054
x1    1   6.3505        28     34.703  0.01173
x2    1   2.4764        27     32.227  0.11557
```

2.3 モデルの評価—疑似 R^2

良いモデルかどうかの評価には，表 3 のように様々な指標がある（L は当該モデルの対数尤度，L_0 はヌル・モデルの対数尤度，p は推定するパラメータ数，n はサンプルサイズ）。

[*6] デビアンスに「推定するパラメータ数の 2 倍」を加えたものが *AIC* である。ヌル・モデルの場合にも，定数項を推定するのでデビアンスに加える数値は 2 である。独立変数が 2 個の場合ならば，定数項と 2 個の独立変数の係数を推定するので $(1+2)\times 2 = 6$ が加えられる

表 3　モデルの評価指標

名前	定義	説明
AIC	$-2L-2p$	値が小さいほど良いモデル
McFadden の R^2	$R^2=1-\dfrac{L_0}{L}$	0 〜 1 の値を取る。1 に近いほど良いモデル
Cox-Snell の R^2	$R^2_{CS}=1-\exp\left\{\dfrac{-2(L-l_0)}{n}\right\}$	McFadden の R^2 の修正版。最大値は 1 ではない
Nagelkerke の R^2	$R^2_N=\dfrac{R^2_{CS}}{1-\exp\left\{\dfrac{2L_0}{n}\right\}}$	Cox-Snell の R^2 の修正版。最大値は 1

本来，R^2（決定係数）は，線形回帰の場合にのみ意味があるものである。ロジスティック回帰などの曲線回帰の場合には，Nagelkerke の疑似 R^2 や Cox & Snell の疑似 R^2 などが計算されることがある[*7]。R においては，ロジスティック回帰を行うには `glm()` を使うのが一般的であるが，`glm()` はこれらの疑似 R^2 を計算しない。`rms` パッケージの `lrm()` は Nagelkerke の疑似 R^2 を計算してくれる。

```
> df <- data.frame(x0=c(52, 43, 54, 40, 40, 54, 44, 56, 65, 53, 39, 62, 52, 55,
+                       48, 49, 56, 40, 28, 57, 59, 47, 50, 71, 43, 59, 37, 40,
+                       52, 60, 58, 52, 50, 44, 71, 50, 56, 48, 46, 40),
+                  y0=c(1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1,
+                       1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0))
> library(rms)
> (lrm.ans <- lrm(y0~x0, df))
Logistic Regression Model

lrm(formula = y0 ~ x0, data = df)

                      Model Likelihood     Discrimination    Rank Discrim.
                         Ratio Test           Indexes          Indexes
Obs            40    LR chi2     11.32    R2       0.329    C       0.787
 0             19    d.f.            1    g        1.539    Dxy     0.574
 1             21    Pr(> chi2) 0.0008    gr       4.661    gamma   0.583
max |deriv| 2e-05                        gp       0.296    tau-a   0.294
                                         Brier    0.188

          Coef    S.E.   Wald Z Pr(>|Z|)
Intercept -7.3360 2.7103 -2.71  0.0068
x0         0.1481 0.0537  2.76  0.0058

> 試験 <- data.frame(
+ 試験結果=factor(rep(1:0, c(11, 15))),
+ 閲覧経験=c(1,1,0,1,0,1,1,0,1,1,1,1,0,0,1,0,1,0,1,1,1,0,0,0,0,1),
+ 勉強時間=c(24,18,15,16,10,26,2,24,18,22,3,6,15,12,6,6,12,12,18,3,8,9,12,6,8,12))
> ans0 <- glm(試験結果~1, data=試験, family=binomial)
> summary(ans0)
Call:
glm(formula = 試験結果 ~ 1, family = binomial, data = 試験)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.049  -1.049  -1.049   1.312   1.312
```

[*7] Cox & Snell の疑似 R^2 は最大値が 1 ではない。これを修正したものが Nagelkerke の疑似 R^2 である。

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.3102     0.3970  -0.781    0.435

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 35.426  on 25  degrees of freedom
Residual deviance: 35.426  on 25  degrees of freedom
AIC: 37.426

Number of Fisher Scoring iterations: 4
> getInfo <- function(obj) {
+         df <- obj$rank
+         aic <- obj$aic
+         ll <- df-aic/2
+         deviance <- -2*ll
+         n <- nrow(obj$data)
+         ll0 <- -obj$null.deviance/2
+         McFaddenR2 <- 1-ll/ll0
+         CoxSnellR2 <- 1-exp(-2*(ll-ll0)/n)
+         NagelkerkeR2 <- CoxSnellR2/(1-exp(2*ll0/n))
+         r <- if (df > 1) cor(obj$y, fitted(obj)) else NA
+         table <- table(obj$y, (fitted(obj) >= 0.5)+0)
+         er <- if (all(dim(table)==c(2, 2))) (table[1,2]+table[2,1])/sum(table) else NA
+         return(list(ll=ll, deviance=deviance, aic=aic, df=df,
+                ll0=ll0, McFaddenR2=McFaddenR2, CoxSnellR2=CoxSnellR2,
+                NagelkerkeR2=NagelkerkeR2, r=r, table=table, er=er))
+ }
> getInfo(ans0)
$ll
[1] -17.71291

$deviance
[1] 35.42582

$aic
[1] 37.42582

$df
[1] 1

$ll0
[1] -17.71291

$McFaddenR2
[1] 2.220446e-16

$CoxSnellR2
[1] 2.220446e-16

$NagelkerkeR2
[1] 2.984518e-16

$r
[1] NA

$table

     0
  0 15
```

```
  1 11

$er
[1] NA
> ans2 <- glm(試験結果~勉強時間, data=試験, family=binomial)
> (info <-getInfo(ans2))
$ll
[1] -14.48565

$deviance
[1] 28.97131

$aic
[1] 32.97131

$df
[1] 2

$ll0
[1] -17.71291

$McFaddenR2
[1] 0.182198

$CoxSnellR2
[1] 0.2198355

$NagelkerkeR2
[1] 0.2954825

$r
[1] 0.5190979

$table

     0  1
  0 13  2
  1  3  8

$er
[1] 0.1923077
> lrt(ans2)
$lr
[1] 6.454512

$df
[1] 1

$p.value
[1] 0.0110671
```

3 多項ロジスティックモデル

多項ロジスティックモデルは，従属変数が 3 つ以上のカテゴリーを持つ名義尺度データに適用できる。従属変数は多項分布に従うと仮定される。

3.1 VGAM パッケージの vglm 関数

3.1.1 使用データと目的

使用するのは iris データセットである。3 種のアヤメ各 50 個の花の 4 個の計測値からアヤメの種を予測することである。

VGAM パッケージの vglm() により，以下のような結果になる。

```
> library(VGAM)
> iris.ans <- vglm(Species ~ ., family=multinomial, data=iris)
> summary(iris.ans)
Call:
vglm(formula = Species ~ ., family = multinomial, data = iris)

Pearson residuals:
                           Min          1Q    Median        3Q       Max
log(mu[,1]/mu[,3]) -2.09e-06 -1.787e-07 4.586e-08 6.329e-08 1.327e-05
log(mu[,2]/mu[,3]) -1.97e+00 -3.382e-04 3.159e-07 4.569e-04 2.560e+00

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept):1     34.243  42494.920   0.001   0.9994
(Intercept):2     42.638     25.708   1.659   0.0972
Sepal.Length:1    10.747  12615.952   0.001   0.9993
Sepal.Length:2     2.465      2.394   1.030   0.3032
Sepal.Width:1     12.815   5841.307   0.002   0.9982
Sepal.Width:2      6.681      4.480   1.491   0.1359
Petal.Length:1   -25.043   8946.662  -0.003   0.9978
Petal.Length:2    -9.429      4.737  -1.990   0.0465
Petal.Width:1    -36.060  14050.767  -0.003   0.9980
Petal.Width:2    -18.286      9.743  -1.877   0.0605

Number of linear predictors:  2

Names of linear predictors: log(mu[,1]/mu[,3]), log(mu[,2]/mu[,3])

Dispersion Parameter for multinomial family:   1

Residual deviance: 11.8985 on 290 degrees of freedom

Log-likelihood: -5.9493 on 290 degrees of freedom

Number of iterations: 21
> ( p <- t(matrix(coefficients(iris.ans), 2)) )
          [,1]       [,2]
[1,]  34.24340  42.637804
[2,]  10.74672   2.465220
[3,]  12.81535   6.680887
[4,] -25.04264  -9.429385
[5,] -36.06029 -18.286137
```

Coefficients:の項に示すように，2 組の切片と傾きの推定値が得られる。

切片と傾きから計算されるものは linear predictor と呼ばれ，判別される群の数より 1 つ少ない個数だけ計算される。

linear predictor は log(mu[,1]/mu[,3]) のように，あるカテゴリー（VGAM では最後のカテゴリー，特に何も指定しないときはアルファベット順で最後のカテゴリーまたは最も大きい整数値が割り当てられるカテゴリー）に対する，別のカテゴリーのロジット（対数オッズ）として表される。

linear predictor は，vglm() が返すオブジェクトを predict() に渡すだけでよい。

```
> lp <- predict(iris.ans)
> head(lp)
  log(mu[,1]/mu[,3]) log(mu[,2]/mu[,3])
1           91.63367           61.73516
2           83.07665           57.90168
3           85.99464           59.68775
4           78.62991           56.88726
5           91.84053           62.15673
6           85.25898           58.66104
```

この結果の意味は，測定値の変数を SL, SW, PL, PW とすると，

$$\begin{cases} setosa \text{ である確率} & = \dfrac{\exp(i_1 + SL_1\, x_1)}{1 + \exp(i_1 + SL_1\, x_1) + \exp(i_2 + SL_2\, x_1)} \\ versicolor \text{ である確率} & = \dfrac{\exp(i_2 + SL_2\, x_1)}{1 + \exp(i_1 + SL_1\, x_1) + \exp(i_2 + SL_2\, x_1)} \\ virginica \text{ である確率} & = \dfrac{1}{1 + \exp(i_1 + SL_1\, x_1) + \exp(i_2 + SL_2\, x_1)} \end{cases} \tag{5}$$

ということである。

胃の内容が，魚，無脊椎動物，その他である確率は表 4 のように推定される。

表 4　54 個の測定値から推定した iris の種の確率

	f1	f2	f3
1	1.0000000	0.0000000	0.0000000
2	1.0000000	0.0000000	0.0000000
3	1.0000000	0.0000000	0.0000000
4	1.0000000	0.0000000	0.0000000
5	1.0000000	0.0000000	0.0000000
51	0.0000000	0.9999883	0.0000117
52	0.0000000	0.9999514	0.0000486
53	0.0000000	0.9988014	0.0011986
54	0.0000000	0.9999578	0.0000422
55	0.0000000	0.9985915	0.0014085
101	0.0000000	0.0000000	1.0000000
102	0.0000000	0.0003861	0.9996139
103	0.0000000	0.0000010	0.9999990
104	0.0000000	0.0002812	0.9997188
105	0.0000000	0.0000001	0.9999999

ここでは，実際に行われる計算過程を示したが，vglm() が返すオブジェクトを fitted() で取り出せばよい。表 5 の確率はこのようにして求めたものであり，当然ながら，表 4 で求めた確率と等しい。

体長から胃の内容を予測するのは，計算された確率が最も高いものであると判別すればよい。判別結果は表 6 に示す。

4　順序ロジスティックモデル

従属変数が 3 つ以上のカテゴリーを持ち，しかもそのカテゴリーに順序関係がある場合（順序尺度データ）に使用されるモデルである。

以下のようなデータを用い g を予測する場合を考えよう。

```
> head(d, 10)
      x    y    z      g
12 31.0 35.4 39.1   poor
```

表 5　4 個の測定値から推定した iris の種

	setosa	versicolor	virginica	predicted
1	1.0000000	0.0000000	0.0000000	setosa
2	1.0000000	0.0000000	0.0000000	setosa
3	1.0000000	0.0000000	0.0000000	setosa
4	1.0000000	0.0000000	0.0000000	setosa
5	1.0000000	0.0000000	0.0000000	setosa
51	0.0000000	0.9999883	0.0000117	versicolor
52	0.0000000	0.9999514	0.0000486	versicolor
53	0.0000000	0.9988014	0.0011986	versicolor
54	0.0000000	0.9999578	0.0000422	versicolor
55	0.0000000	0.9985915	0.0014085	versicolor
101	0.0000000	0.0000000	1.0000000	virginica
102	0.0000000	0.0003861	0.9996139	virginica
103	0.0000000	0.0000010	0.9999990	virginica
104	0.0000000	0.0002812	0.9997188	virginica
105	0.0000000	0.0000001	0.9999999	virginica

表 6　判別結果

	setosa	versicolor	virginica	合計
setosa	50	0	0	50
versicolor	0	49	1	50
virginica	0	1	49	50
合計	50	50	50	150

```
39 32.5 36.3 37.7   poor
7  32.9 39.3 36.0   poor
3  36.8 29.2 31.9   poor
1  38.1 43.5 48.1   poor
25 39.0 36.8 32.1   poor
38 39.0 44.1 45.0   poor
52 40.0 44.4 48.1 normal
11 40.4 55.9 49.8   poor
8  41.2 58.3 48.4   poor
```

g は，“poor” < “normal” < “good” のように順序がついている。

累積ロジスティックモデル（cumulative logistic model）では，まず，“poor” を「反応なし」，“normal” と “good” を「反応あり」として

$$\lambda_1 = \log \frac{\pi_1}{1-\pi_1} = \beta_{10} + \beta_{11}\, x_1 + \cdots + \beta_{1p}\, x_p + \varepsilon_1 \tag{6}$$

を当てはめる。

次に “poor” と “normal” を「反応なし」，“good” を「反応あり」として

$$\lambda_2 = \log \frac{\pi_2}{1-\pi_2} = \beta_{20} + \beta_{21}\, x_1 + \cdots + \beta_{2p}\, x_p + \varepsilon_2 \tag{7}$$

を当てはめる。

ここで，(6) 式と (7) 式の偏回帰係数において，$\beta_{10} \neq \beta_{20}$ であるがそのほかの偏回帰係数は $\beta_{1i} = \beta_{2i}$ であると考える場合と，偏回帰係数は全て異なる（$\beta_{1i} \neq \beta_{2i}$）と考える場合とがある。

前者は，2 つのモデルのオッズの間には比例関係がありその比例定数は $\exp(\beta_{10} - \beta_{20})$ となるので，このような累積ロジスティックモデルは特に比例オッズモデル（proportional odds model, POM）と呼ぶ。2 つのロジスティック曲線は図 3 のように平行移動したものになる。偏回帰係数が同じということは，2 つのモデルのロ

ジットつまり対数オッズの違いは，定数項の差に影響されるだけで，説明変数には影響されないということである。2 つのモデルのオッズの間には比例関係があり，その比例定数は定数項の差を指数変換した値になる。

後者では図 4 のように，2 つのロジスティック曲線の形状は異なる。

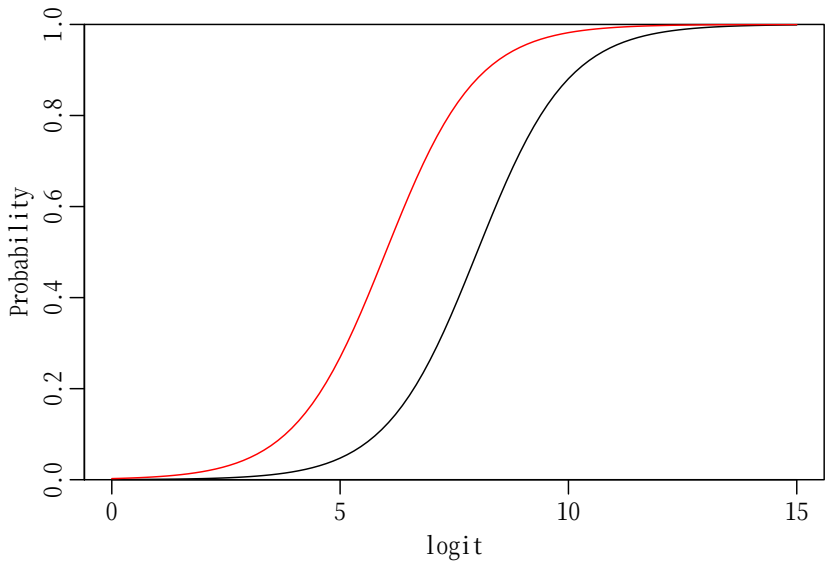

図 3　比例オッズモデル

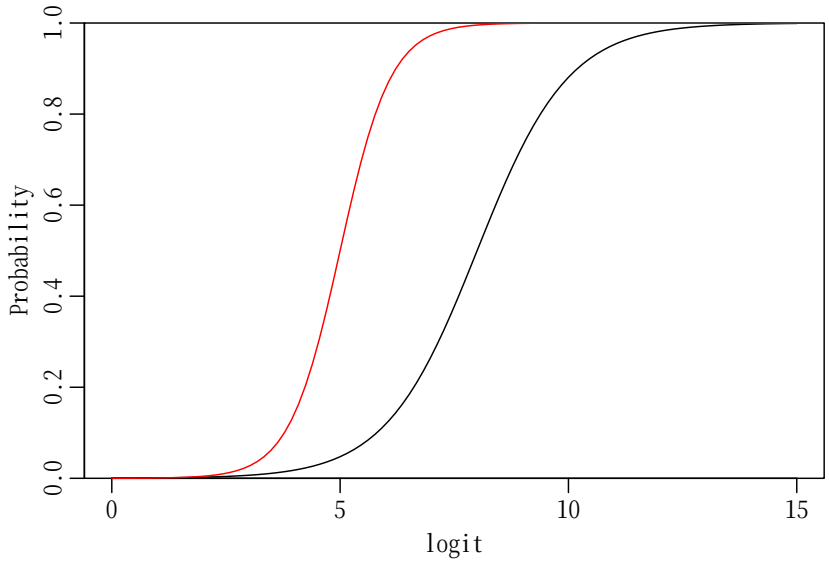

図 4　累積ロジスティックモデル

図 5 の黒と赤で表されたロジスティック曲線が，それぞれ（6）式と（7）式によるものとする。ロジットが x の場合そのデータが従属変数のどのカテゴリーに属するかは，以下のように予測される。

まず，赤のロジスティック曲線は “normal” と “good” を「反応あり」としたものであるから，確率 $p2$ は “normal” または “good”である確率なので，$1 - p2$” が “poor” の確率である。

次に，黒のロジスティック曲線は “good” を「反応あり」としたものであるから，確率 $p1$ は “good”である確率である。

よって，$1 - \{(1 - p2) + p1\} = p2 - p1$ が “normal” である確率である。

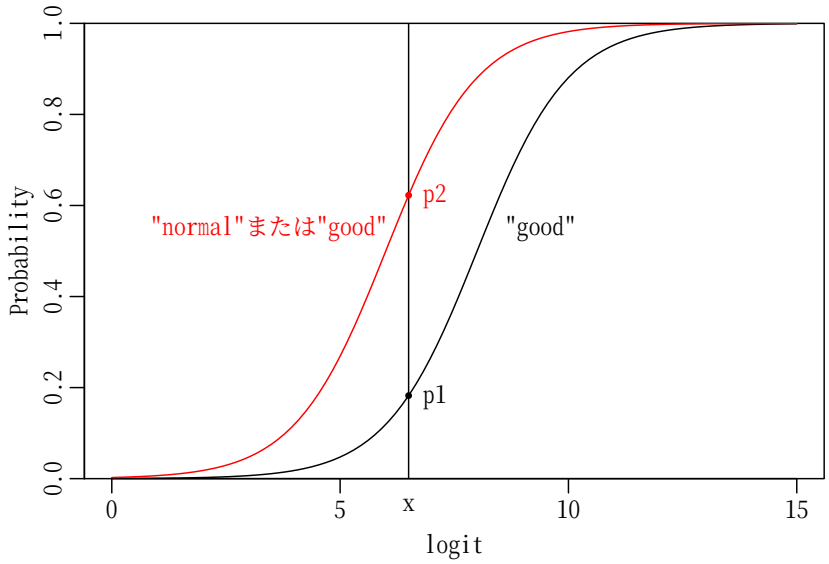

図 5　それぞれの確率

4.1　VGAM パッケージの vglm 関数

4.1.1　累積ロジスティックモデル parallel=FALSE

累積ロジスティックモデルを当てはめるには，vglm() において parallel=FALSE を指定する。

```
> library(VGAM)
> ans1 <- vglm(g ~., family=cumulative(parallel=FALSE), data=d)
> summary(ans1)
Call:
vglm(formula = g ~ ., family = cumulative(parallel = FALSE),
    data = d)

Pearson residuals:
                    Min      1Q  Median     3Q   Max
logit(P[Y<=1]) -2.172 -0.5420 -0.2174 0.5755 3.807
logit(P[Y<=2]) -2.916 -0.6093  0.2114 0.6013 2.623

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept):1  7.23426    1.53041   4.727 2.28e-06
(Intercept):2  8.76940    1.68852   5.194 2.06e-07
x:1           -0.05812    0.03193  -1.820   0.0687
x:2           -0.05553    0.03124  -1.778   0.0755
y:1           -0.05573    0.03806  -1.464   0.1431
y:2           -0.05483    0.03788  -1.447   0.1478
z:1           -0.02682    0.04308  -0.623   0.5336
z:2           -0.02222    0.04206  -0.528   0.5972

Number of linear predictors:  2

Names of linear predictors: logit(P[Y<=1]), logit(P[Y<=2])

Dispersion Parameter for cumulative family:   1

Residual deviance: 207.9834 on 232 degrees of freedom

Log-likelihood: -103.9917 on 232 degrees of freedom
```

```
Number of iterations: 5
```

係数は coefficients() によって取り出すことができる。

```
> coeff <- matrix(coefficients(ans1), byrow=TRUE, ncol=2)
> dimnames(coeff) <- list(c("Intercept", "x", "y", "z"),
+                         paste("logit", 1:2))
> coeff
              logit 1     logit 2
Intercept  7.23426154  8.76939524
x         -0.05811895 -0.05552564
y         -0.05573438 -0.05482532
z         -0.02681610 -0.02222356
```

それぞれのロジットは次のようになる。

```
> logit <- predict(ans1)
> head(logit)
   logit(P[Y<=1]) logit(P[Y<=2])
12       2.411068       4.238343
39       2.311271       4.136824
7        2.166407       3.987918
3        2.612607       4.416221
1        1.305630       3.200013
25       2.055801       3.872947
```

logit は以下のように計算されている。

```
> head(t(t(as.matrix(d[, 1:3]) %*% coeff[2:4,])+coeff[1,])) # = logit
    logit 1  logit 2
12 2.411068 4.238343
39 2.311271 4.136824
7  2.166407 3.987918
3  2.612607 4.416221
1  1.305630 3.200013
25 2.055801 3.872947
```

例えば，1 番目のデータは

```
> d[1,]
    x    y    z    g
12 31 35.4 39.1 poor
```

なので，係数を掛けて足し合わせ，切片を加えて以下のように計算される。

```
logit(P[Y <= 1])= 7.23426153807201 + -0.0581189503059378 × 31 + -0.0557343811280323
× 35.4 + -0.026816095193425 × 39.1 = 2.411068
logit(P[Y <= 2])= 8.76939523800393 + -0.0555256446077885 × 31 + -0.05482532423325
× 35.4 + -0.0222235593208556 × 39.1 = 4.238343
```

従属変数のそれぞれのカテゴリーに対応する確率を求めるには，predict() で type="response"を指定すればよい。

```
> p <- predict(ans1, type="response")
> head(p)
        poor     normal       good
12 0.9176674 0.06810643 0.01422619
39 0.9098062 0.07447145 0.01572236
7  0.8971921 0.08460708 0.01820086
3  0.9316685 0.05639584 0.01193562
1  0.7867809 0.17405384 0.03916522
25 0.8865324 0.09309429 0.02037329
```

これは，次のように計算されている。確率 P[Y<=1]，P[Y<=2] は 1/({1 + exp(−*logit*)} で計算される。

```
> P <- 1/(1+exp(-logit))
> colnames(P) <- c("P[Y<=1]", "P[Y<=2]")
> head(P)
     P[Y<=1]   P[Y<=2]
12 0.9176674 0.9857738
39 0.9098062 0.9842776
7  0.8971921 0.9817991
3  0.9316685 0.9880644
1  0.7867809 0.9608348
25 0.8865324 0.9796267
```

これに基づいて，確率 P[Y=1]，P[Y=2]，P[Y=3] は以下のように計算される。

```
> P2 <- cbind(P[,1], P[,2]-P[,1], 1-P[,2])
> colnames(P2) <- paste("P[Y=", 1:3, "]", sep="")
> head(P2)
      P[Y=1]     P[Y=2]     P[Y=3]
12 0.9176674 0.06810643 0.01422619
39 0.9098062 0.07447145 0.01572236
7  0.8971921 0.08460708 0.01820086
3  0.9316685 0.05639584 0.01193562
1  0.7867809 0.17405384 0.03916522
25 0.8865324 0.09309429 0.02037329
```

また，元のデータでの確率は fitted() で求めることができる。

```
> head(fitted(ans1))
        poor     normal       good
12 0.9176674 0.06810643 0.01422619
39 0.9098062 0.07447145 0.01572236
7  0.8971921 0.08460708 0.01820086
3  0.9316685 0.05639584 0.01193562
1  0.7867809 0.17405384 0.03916522
25 0.8865324 0.09309429 0.02037329
```

判別結果は以下のようになる。

```
> res <- apply(p, 1, which.max)
> xtabs(~ d$g + res)
        res
d$g       1  2  3
  poor   24 14  2
  normal 11 19 10
  good    4 11 25
```

4.1.2 累積ロジスティックモデル parallel=TRUE

比例オッズモデルを当てはめるには，vglm() において parallel=TRUE を指定する。

```
> ans.vglm <- vglm(g ~ ., family=cumulative(parallel=TRUE), data=d)
> summary(ans.vglm)
Call:
vglm(formula = g ~ ., family = cumulative(parallel = TRUE), data = d)

Pearson residuals:
                   Min      1Q  Median     3Q   Max
logit(P[Y<=1]) -2.100 -0.5609 -0.2214 0.5914 3.682
logit(P[Y<=2]) -3.003 -0.5971  0.2065 0.5812 2.726
```

```
Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept):1  7.01710    1.23920   5.663 1.49e-08
(Intercept):2  9.02639    1.35863   6.644 3.06e-11
x             -0.05682    0.02572  -2.209   0.0272
y             -0.05539    0.03096  -1.789   0.0736
z             -0.02436    0.03471  -0.702   0.4829

Number of linear predictors:  2

Names of linear predictors: logit(P[Y<=1]), logit(P[Y<=2])

Dispersion Parameter for cumulative family:   1

Residual deviance: 208.0519 on 235 degrees of freedom

Log-likelihood: -104.026 on 235 degrees of freedom

Number of iterations: 5
```

ロジットは以下のように計算できる。

```
> logit <- predict(ans.vglm) # type="link"
> head(logit)
   logit(P[Y<=1]) logit(P[Y<=2])
12       2.342324       4.351615
39       2.241334       4.250626
7        2.093839       4.103131
3        2.531534       4.540825
1        1.270986       3.280277
25       1.980677       3.989968
```

この計算は，以下のように行われている。

```
> coeff <- coefficients(ans.vglm)
> coeff
(Intercept):1 (Intercept):2             x             y             z
   7.01710005    9.02639171   -0.05682499   -0.05539087   -0.02435716
> logit1 <- t(t(as.matrix(d[, 1:3]) %*% coeff[3:5])+coeff[1]) # = logit1
> logit2 <- t(t(as.matrix(d[, 1:3]) %*% coeff[3:5])+coeff[2]) # = logit2
> head(cbind(logit1, logit2))
       [,1]     [,2]
12 2.342324 4.351615
39 2.241334 4.250626
7  2.093839 4.103131
3  2.531534 4.540825
1  1.270986 3.280277
25 1.980677 3.989968
```

従属変数の各カテゴリーに対する確率は次のように計算できる。

```
> p <- predict(ans.vglm, type="response")
> head(p)
         poor     normal       good
12 0.9123221 0.07495582 0.01272205
39 0.9039004 0.08204462 0.01405495
7  0.8903029 0.09344472 0.01625237
3  0.9263231 0.06312484 0.01055207
1  0.7809114 0.18283454 0.03625402
25 0.8787533 0.10308248 0.01816426
```

この計算は，以下のように行われている。

```
> P <- 1/(1+exp(-logit))
> colnames(P) <- c("P[Y<=1]", "P[Y<=2]")
> head(P)
     P[Y<=1]   P[Y<=2]
12 0.9123221 0.9872780
39 0.9039004 0.9859451
7  0.8903029 0.9837476
3  0.9263231 0.9894479
1  0.7809114 0.9637460
25 0.8787533 0.9818357
> P2 <- cbind(P[,1], P[,2]-P[,1], 1-P[,2])
> colnames(P2) <- paste("P[Y=", 1:3, "]", sep="")
> head(P2)
      P[Y=1]     P[Y=2]     P[Y=3]
12 0.9123221 0.07495582 0.01272205
39 0.9039004 0.08204462 0.01405495
7  0.8903029 0.09344472 0.01625237
3  0.9263231 0.06312484 0.01055207
1  0.7809114 0.18283454 0.03625402
25 0.8787533 0.10308248 0.01816426
```

また，元のデータでの確率は fitted() で求めることができる。

```
> head(fitted(ans.vglm))
        poor     normal       good
12 0.9123221 0.07495582 0.01272205
39 0.9039004 0.08204462 0.01405495
7  0.8903029 0.09344472 0.01625237
3  0.9263231 0.06312484 0.01055207
1  0.7809114 0.18283454 0.03625402
25 0.8787533 0.10308248 0.01816426
```

判別結果は以下のようになる。

```
> res <- apply(p, 1, which.max)
> xtabs(~ d$g + res)
        res
d$g       1  2  3
  poor   24 14  2
  normal 11 19 10
  good    4 11 25
```

4.2 MASS パッケージの polr 関数

MASS パッケージの polr() は比例オッズモデルである。

独立変数に対する回帰係数は，VGAM パッケージの vglm() とは符号が逆になっていることと回帰係数の精度が少し異なることに注意が必要である。

```
> library(MASS)
> ans.polr <- polr(g ~., d, Hess=TRUE)
> summary(ans.polr)
Call:
polr(formula = g ~ ., data = d, Hess = TRUE)

Coefficients:
    Value Std. Error t value
x 0.05683    0.02543  2.2342
y 0.05539    0.03075  1.8015
z 0.02436    0.03478  0.7002
```

```
Intercepts:
            Value  Std. Error t value
poor|normal 7.0171 1.2321     5.6952
normal|good 9.0264 1.3521     6.6759

Residual Deviance: 208.0519
AIC: 218.0519
```

polr() が返すオブジェクトの中には，ロジットが含まれていないが，ロジットは以下のように計算できる。

```
> coeff <- -ans.polr$coefficients # = -coefficients(ans.polr)
> const <- ans.polr$zeta
> logit <- cbind(as.matrix(d[, 1:3]) %*% coeff+const[1], as.matrix(d[, 1:3]) %*% coeff+const[2])
> head(logit)
       [,1]     [,2]
12 2.342323 4.351615
39 2.241334 4.250626
7  2.093839 4.103131
3  2.531532 4.540824
1  1.270985 3.280277
25 1.980676 3.989968
```

従属変数の各カテゴリーに対する確率は次のように計算できる。

```
> P <- 1/(1+exp(-logit))
> head(P)
        [,1]      [,2]
12 0.9123221 0.9872780
39 0.9039004 0.9859450
7  0.8903029 0.9837476
3  0.9263230 0.9894479
1  0.7809114 0.9637460
25 0.8787532 0.9818357
> P2 <-t(apply(P, 1, function(x) diff(c(0, x, 1))))
> head(P2)
        [,1]       [,2]       [,3]
12 0.9123221 0.07495586 0.01272205
39 0.9039004 0.08204466 0.01405495
7  0.8903029 0.09344474 0.01625237
3  0.9263230 0.06312491 0.01055208
1  0.7809114 0.18283460 0.03625403
25 0.8787532 0.10308253 0.01816426
```

実際には，predict() で type="probs"を指定すれば簡単に求めることができる。

```
> p <- predict(ans.polr, type="probs")
> head(p)
        poor     normal       good
12 0.9123221 0.07495586 0.01272205
39 0.9039004 0.08204466 0.01405495
7  0.8903029 0.09344474 0.01625237
3  0.9263230 0.06312491 0.01055208
1  0.7809114 0.18283460 0.03625403
25 0.8787532 0.10308253 0.01816426
```

データが，従属変数のどのカテゴリーに属するかの予測は predict(ans.polr) によって求まるので，実際のカテゴリーと比較して予測の正確性を検討できる。

```
> prediction.polr <- predict(ans.polr)
> head(prediction.polr)
[1] poor poor poor poor poor poor
```

```
Levels: poor normal good
> table(d$g, prediction.polr)
        prediction.polr
         poor normal good
  poor     24     14    2
  normal   11     19   10
  good      4     11   25
```

また，元のデータでの確率は fitted() で求めることができる。

```
> head(fitted(ans.polr))
        poor     normal       good
12 0.9123221 0.07495586 0.01272205
39 0.9039004 0.08204466 0.01405495
7  0.8903029 0.09344474 0.01625237
3  0.9263230 0.06312491 0.01055208
1  0.7809114 0.18283460 0.03625403
25 0.8787532 0.10308253 0.01816426
```

判別結果は以下のようになる。

```
> res <- apply(p, 1, which.max)
> xtabs(~ d$g + res)
        res
d$g       1  2  3
  poor   24 14  2
  normal 11 19 10
  good    4 11 25
```