Multivariate Verfahren

MSc Psychologie | MSc Klinische Psychologie und Psychotherapie

WiSe 2023/24

Prof. Dr. Dirk Ostwald

(12) Logistische Regression

Anwendungsszenario

Modellformulierung

Klassifikation

Lernen

Anwendungsbeispiel

Selbstkontrollfragen

Anwendungsszenario

Psychotherapie Non-Response-Rate wird auf etwa 20 - 30% geschätzt

Vorhersage von Behandlungserfolg basierend auf klinischen Markern wäre hilfreich

- Therapieauswahloptimierung
- Lebensqualitätverbesserung
- Ressourcensensitivität

Digitale Datenbank von Psychotherapieverläufen als Trainingsdatensatz

Prädiktive Modellierung zur Etablierung eines prädiktiven klinischen Markerprofils

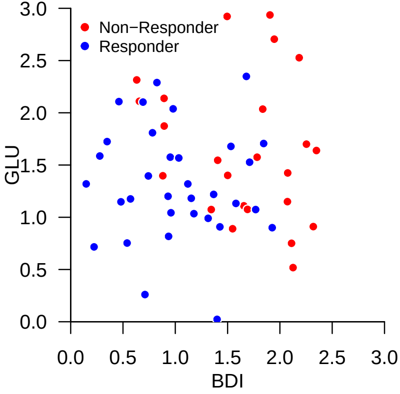
Treatmentsuccessvorhersage für neue Patient:innen

Anwendungsbeispiele

- BDI und GLU Werte bei Depressionsdiagnose als Prädiktoren von CBT Erfolg
- Lineare Diskriminanzanalyse, Logistische Regression, Neuronale Netze

Anwendungsszenario

BDI und GLU Werte bei Depressionsdiagnose als Prädiktoren von CBT Erfolg



Anwendungsszenario

Modellformulierung

Klassifikation

Lernen

Anwendungsbeispiel

Selbstkontrollfragen

Definition (Generalisiertes Lineares Modell)

$x \in \mathbb{R}^{m+1}$ sei ein erweiterter Featurevektor und υ das assoziierte Label. Weiterhin sei für einen *Parametervektor* $\beta \in \mathbb{R}^{m+1}$

$$\eta := x^T\beta \tag{1}$$

ein *linearer Prädiktor*. Dann ist ein generalisierte lineares Modell definiert mithilfe einer zweimal differenzierbaren und invertierbaren *Link-Funktion*

$$g : \mathbb{R} \to \mathbb{R}, \mathbb{E}(\upsilon) \mapsto g(\mathbb{E}(\upsilon)) =: \eta. \tag{2}$$

definiert. Die Inverse der Link-Funktion,

$$g^{-1} : \mathbb{R} \to \mathbb{R}, \eta \mapsto g^{-1}(\eta) = \mathbb{E}(\upsilon) \tag{3}$$

heißt *Mean-Funktion* und wird mit f bezeichnet, so dass

$$f : \mathbb{R} \to \mathbb{R}, \eta \mapsto f(\eta) = \mathbb{E}(\upsilon). \tag{4}$$

Definition (ALM als Generalisiertes Lineares Modell)

Das Allgemeine Lineare Modell mit u.i.v. Störvariablen ist das Generalisierte Lineare Model, bei dem

1. die Labelvariable eine univariat normalverteilte Zufallsvariable

$$\upsilon \sim N(\mu, \sigma^2), \tag{5}$$

   ist und

2. die Link-Funktion durch die Identität

$$g : \mathbb{R} \to \mathbb{R}, \mu \mapsto g(\mu) := \mu =: \eta. \tag{6}$$

   gegeben ist.

Weil die Inverse der Identität wiederrum die Identität ist, folgt, dass die Mean-Funktion des ALM durch

$$f : \mathbb{R} \to \mathbb{R}, \eta \mapsto f(\eta) = \eta = \mu. \tag{7}$$

gegeben ist. Die Parameter des Allgemeinen Linearen Modells sind die Komponenten des Vektors $\beta \in \mathbb{R}^m$ des linearen Prädiktors $\eta = x^T\beta$ und der Parameter $\sigma^2 > 0$.

Definition (Logistische Regression als Generalisiertes Lineares Modell)

Das Modell der Logistischen Regression (LR) ist das Generalisierte Lineare Modell, bei dem

1. die Labelvariable eine Bernoulli-Zufallsvariable

$$\upsilon \sim B(\mu) \tag{8}$$

   ist und

2. die Link-Funktion durch die *standard logit function*

$$g : [0,1] \to \mathbb{R}, \mu \mapsto g(\mu) := \ln\left(\frac{\mu}{1-\mu}\right) =: \eta \tag{9}$$

   gegeben ist.

Die Parameter des Logistischen Regressionsmodels sind die Komponenten des Vektors $\beta \in \mathbb{R}^m$ des linearen Prädiktors $\eta = x^T\beta$.

Theorem (Mean-Funktion der Logistischen Regression)

Die Inverse der Link-Funktion des Modells der Logistischen Regression und somit seine Mean-Funktion ist die *Logistische Standardfunktion*

$$f : \mathbb{R} \to [0,1], \eta \mapsto f(\eta) = \frac{1}{1+\exp(-\eta)}. \tag{10}$$

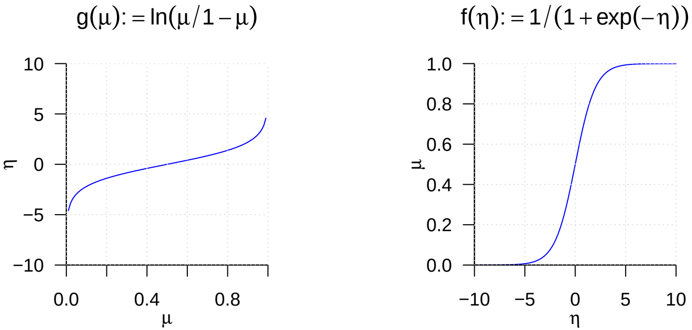
Beweis

Umformen der logit function ergibt

$$\begin{aligned}
\eta &= \ln(\mu/(1-\mu)) \\
\Leftrightarrow -\eta &= -\ln(\mu/(1-\mu)) \\
\Leftrightarrow -\eta &= \ln((1-\mu)/\mu)) \\
\Leftrightarrow \exp(-\eta) &= (1-\mu)/\mu \\
\Leftrightarrow \mu\exp(-\eta) &= 1-\mu \\
\Leftrightarrow \exp(-\eta) &= \mu^{-1}-1 \\
\mu &= 1/(\exp(-\eta)+1)
\end{aligned}$$

$\square$

Link und Mean Funktionen



Definition (Modell der Logistischen Regression)

υ sei eine Zufallsvariable mit Ergebnisraum $\{0, 1\}$. Dann ist das *Modell der Logistischen Regressions* definiert als die WMF von υ

$$p(y) = B\left(y; \frac{1}{1+\exp(-x^T\beta)}\right), \tag{11}$$

wobei $x \in \mathbb{R}^{m+1}$ einen erweiterten Featurevektor und $\beta \in \mathbb{R}^{m+1}$ den *Parametervektor* bezeichnen

Bemerkung

- Aus generativer Sicht wird ein Trainingsdatensatz

$$\left\{\left(x^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n} \text{ mit } x^{(i)} \in \mathbb{R}^{m+1} \text{ und } y^{(i)} \in \{0,1\} \tag{12}$$

eines LR Modells wie folgt erzeugt:

(1) Definition von $x^{(i)}$,
(2) Ziehen von $y^{(i)}$ aus $p(y) = B(y; \mu)$ mit Erwartungswertparameter $\mu = \frac{1}{1+\exp(-x^{(i)T}\beta)}$.

Modellformulierung

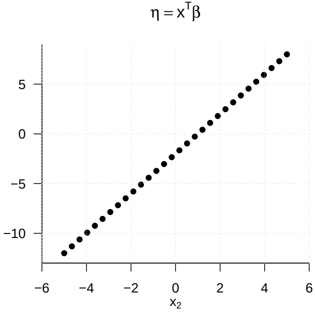
Datengeneration bei einfacher Logistischer Regression für $m = 1, \beta = (-2, 2)^T, n = 30$.

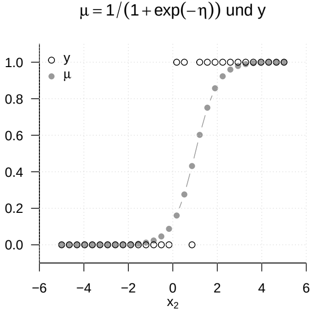
```
# Modellparameter
m    = 1                                # Featurevektoredimensionalität
n    = 30                               # Anzahl Datenpunkte
x    = matrix(c(rep(1,n),               # Definition des erweiterten Featurevektors
                seq(-5,5, len = n)),
              nrow  = 2,
              byrow = TRUE)
beta = matrix(c(-2,2), nrow = 2)        # wahrer, aber unbekannter, Parametervektor
eta  = t(x) %*% beta                    # wahrer, aber unbekannter linearer Prädiktor
mu   = 1/(1+exp(-eta))                  # wahrer, aber unbekannter, Bernoulliparametervektor

# Dategeneration
set.seed(2)                             # Zufallsgeneratorzustand
y    = rep(NaN,2)                       # Datenarray
for(i in 1:n){
  y[i] = rbinom(1,1,mu[i])              # Bernoullivariablenrealisierung
}
```

Modellformulierung

Datengeneration bei einfacher Logistischer Regression für $m = 1, \beta = (-2, 2)^T, n = 30$.





Definition (Klassifikationsregel der Logistischen Regression)

$p(y)$ sei die WMF eines Logistischen Regressionsmodells. Dann ist die *Klassifikationregel* definiert als

$$\delta : \mathbb{R}^m \to \{0,1\}, x \mapsto \delta(x) := \begin{cases} 0 & \text{für } p(y=0) \geq p(y=1) \\ 1 & \text{für } p(y=0) < p(y=1) \end{cases} \tag{13}$$

Bemerkung

- Es gilt

$$\delta(x) = 1 \Leftrightarrow p(y=1) > p(y=0) \Leftrightarrow p(y=1) > 0.5. \tag{14}$$

Anwendungsszenario

Modellformulierung

Klassifikation

Lernen

Anwendungsbeispiel

Selbstkontrollfragen

Theorem (Log-Likelihood-Funktion der Logistischen Regression)

$\{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$ sei ein Trainingsdatensatz aus erweiterten Featurevektoren und assoziierten Labelvariablenrealisierungen und f sei die Logistische Standardfunktion. Dann hat die Log-Likelihood-Funktion der Logistischen Regression die Form

$$\ell : \mathbb{R}^m \to \mathbb{R}, \beta \mapsto \ell(\beta) := \sum_{i=1}^{n} y^{(i)} \ln(f({x^{(i)}}^T\beta)) + (1 - y^{(i)}) \ln(1 - f({x^{(i)}}^T\beta)).$$

Beweis

Wir halten zunächst fest, dass für u.i.v. Labelvariablen gilt, dass

$$\ell(\beta) := \ln p(y^{(1)}, ..., y^{(n)}) = \ln \prod_{i=1}^{n} p(y^{(i)}) = \sum_{i=1}^{n} \ln p(y^{(i)}) \tag{15}$$

Mit der WMF der Bernoulliverteilung folgt dann

$$\begin{aligned} \ell(\beta) &= \sum_{i=1}^{n} \ln \left(f\left({x^{(i)}}^T\beta\right)^{y^{(i)}} \left(1 - f\left({x^{(i)}}^T\beta\right)\right)^{1-y^{(i)}} \right) \\ &= \sum_{i=1}^{n} y^{(i)} \ln\left(f\left({x^{(i)}}^T\beta\right)\right) + \left(1 - y^{(i)}\right) \ln\left(1 - f\left({x^{(i)}}^T\beta\right)\right) \end{aligned} \tag{16}$$

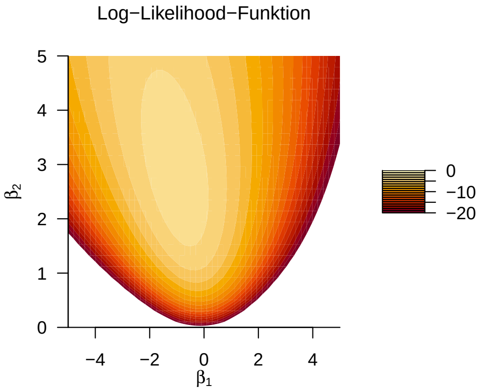
Log-Likelihood-Funktion des generierten Datensatzes für $m = 1, \beta = (-2, 2)^T, n = 30$.

```
# Funktionsdefinitionen
# -----------------------------------------------------------------------------------------------
# Standard Logistic Function
f = function(eta){
    return(1/(1 + exp(-eta)))
}

# Log Likelihood Function
llh = function(x,y,beta){
    n   = ncol(x)
    ell = 0
    for(i in 1:n){
        ell = ell + y[i]*log(f(t(x[,i]) %*% beta)) + (1-y[i])*log(1-f(t(x[,i]) %*% beta))
    }
    return(ell)
}

# Log-Likelihood-Funktion Auswertung
# -----------------------------------------------------------------------------------------------
beta_min  = -5                                                      # beta Minimum
beta_max  =  5                                                      # beta Maximum
beta_res  = 5e1                                                     # beta Auflösung
beta_1    = seq(beta_min, beta_max, length.out = beta_res)          # beta_1 Raum
beta_2    = seq(beta_min, beta_max, length.out = beta_res)          # beta_2 Raum
ell       = matrix(rep(NaN, beta_res*beta_res), nrow = beta_res) # Log-Likelihood-Funktion Array
for(i in 1:beta_res){
    for(j in 1:beta_res){
        beta12   = matrix(c(beta_1[i], beta_2[j]), nrow = 2)
        ell[i,j] = llh(x,y,beta12)
    }
}
```

Log-Likelihood-Funktion des generierten Datensatzes für $m = 1, \beta = (-2, 2)^T, n = 30$.



Theorem (Gradientverfahren der Logistischen Regression)

Gegeben sei das Modell einer Logistischen Regression und $\{(x^{(i)}, y^{(i)})\}_{i=1}^n$ sei ein entsprechender Trainingsdatensatz. Dann kann ein Maximum-Likelihood-Schätzer $\hat{\beta}$ für den Parametervektor β des Logistischen Regressionsmodell durch folgendes Gradientenverfahren gewonnen werden:

(0) Wähle $\beta^0 \in \mathbb{R}, \alpha > 0, \delta > 0$

(1) Für $k = 0, 1, 2, \ldots$ bis zur Konvergenz setze

$$\beta^{(k+1)} := \beta^{(k)} + \alpha \nabla \ell(\beta^{(k)}). \tag{17}$$

wobei $\nabla \ell \left(\beta^k\right)$ den Gradienten der Log-Likelihood-Funktion der Logistischen Regression bezeichnet und die Form

$$\nabla \ell(\beta) = \begin{pmatrix} \frac{\partial}{\partial \beta_1} \ell(\beta) \\ \frac{\partial}{\partial \beta_2} \ell(\beta) \\ \vdots \\ \frac{\partial}{\partial \beta_m} \ell(\beta) \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n (y^{(i)} - f({x^{(i)}}^T \beta)) x_1^{(i)} \\ \sum_{i=1}^n (y^{(i)} - f({x^{(i)}}^T \beta)) x_2^{(i)} \\ \vdots \\ \sum_{i=1}^n (y^{(i)} - f({x^{(i)}}^T \beta)) x_m^{(i)} \end{pmatrix} \tag{18}$$

Bemerkungen

- Das reine Gradientenverfahren zum Lernen der Parameter eines LR Modells ist recht instabil.
- Iteratively Weighted Least Squares Verfahren werden zur ML Schätzung in GLMs bevorzugt (Green (1984)).
- IWLS Verfahren nutzen Gradienten und Hesse-Matrix ähnlich wie Gauss-Newton Verfahren.
- R implementiert in der `glm()` ein IWLS Verfahren.

Lernen

Beweis

Um die $j$te partielle Ableitung der Log-Likelihood-Funktion zu bestimmen, halten wir zunächst fest, dass sich die Ableitung der logistic function f hinsichtlich η zu

$$\frac{d}{d\eta}f : \mathbb{R} \to \mathbb{R}, \eta \mapsto \frac{d}{d\eta}f(\eta) = f(\eta)(1 - f(\eta)) \tag{19}$$

ergibt. Dies kann wie folgt eingesehen werden:

$$\begin{aligned}
\frac{d}{d\eta}f(\eta) &= \frac{d}{d\eta}\left(1 + \exp(-\eta)\right)^{-1} \\
&= -\left(1 + \exp(-\eta)\right)^{-2} \cdot \exp(-\eta) \cdot (-1) \\
&= \frac{\exp(-\eta)}{(1 + \exp(-\eta))^2} \\
&= \frac{1 + \exp(-\eta) - 1}{(1 + \exp(-\eta))^2} \\
&= \frac{1 + \exp(-\eta)}{(1 + \exp(-\eta))^2} - \frac{1}{(1 + \exp(-\eta))^2} \\
&= \frac{1}{1 + \exp(-\eta)} - \frac{1}{(1 + \exp(-\eta))^2} \\
&= \frac{1}{1 + \exp(-\eta)}\left(1 - \frac{1}{1 + \exp(-\eta)}\right) \\
&= f(\eta)(1 - f(\eta))
\end{aligned}$$

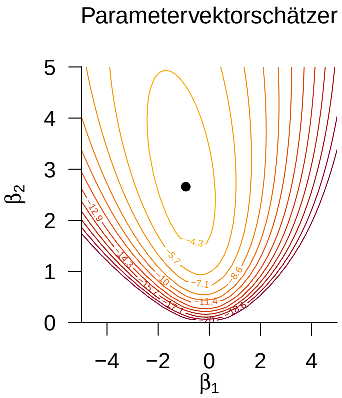
Damit ergibt sich dann für $\frac{\partial}{\partial \beta_j}\ell, j = 1, ..., m$:

Lernen

$$
\begin{aligned}
\frac{\partial}{\partial \beta_j}\ell(\beta) &= \frac{\partial}{\partial \beta_j}\left(\sum_{i=1}^n y^{(i)} \ln\left(f\left({x^{(i)}}^T\beta\right)\right) + \left(1-y^{(i)}\right)\ln\left(1-f\left({x^{(i)}}^T\beta\right)\right)\right) \\
&= \sum_{i=1}^n y^{(i)} \frac{\partial}{\partial \beta_j}\left(\ln\left(f\left({x^{(i)}}^T\beta\right)\right)\right) + \left(1-y^{(i)}\right)\frac{\partial}{\partial \beta_j}\left(\ln\left(1-f\left({x^{(i)}}^T\beta\right)\right)\right) \\
&= \sum_{i=1}^n y^{(i)} \frac{1}{f\left({x^{(i)}}^T\beta\right)}\left(\frac{\partial}{\partial \beta_j}\left(f\left({x^{(i)}}^T\beta\right)\right)\right) + \left(1-y^{(i)}\right)\frac{1}{1-f\left({x^{(i)}}^T\beta\right)}\frac{\partial}{\partial \beta_j}\left(1-f\left({x^{(i)}}^T\beta\right)\right) \\
&= \sum_{i=1}^n \left(y^{(i)} \frac{1}{f\left({x^{(i)}}^T\beta\right)} - \left(1-y^{(i)}\right)\frac{1}{1-f\left({x^{(i)}}^T\beta\right)}\right)\frac{\partial}{\partial \beta_j}\left(f\left({x^{(i)}}^T\beta\right)\right) \\
&= \sum_{i=1}^n \left(y^{(i)} \frac{1}{f\left({x^{(i)}}^T\beta\right)} - \left(1-y^{(i)}\right)\frac{1}{1-f\left({x^{(i)}}^T\beta\right)}\right) f\left({x^{(i)}}^T\beta\right)\left(1-f\left({x^{(i)}}^T\beta\right)\right)\frac{\partial}{\partial \beta_j}\left({x^{(i)}}^T\beta\right) \\
&= \sum_{i=1}^n \left(y^{(i)} \frac{1}{f\left({x^{(i)}}^T\beta\right)} - \left(1-y^{(i)}\right)\frac{1}{1-f\left({x^{(i)}}^T\beta\right)}\right) f\left({x^{(i)}}^T\beta\right)\left(1-f\left({x^{(i)}}^T\beta\right)\right) x_j^{(i)} \\
&= \sum_{i=1}^n \left(y^{(i)} \frac{f\left({x^{(i)}}^T\beta\right)\left(1-f\left({x^{(i)}}^T\beta\right)\right)}{f\left({x^{(i)}}^T\beta\right)} - \left(1-y^{(i)}\right)\frac{f\left({x^{(i)}}^T\beta\right)\left(1-f\left({x^{(i)}}^T\beta\right)\right)}{1-f\left({x^{(i)}}^T\beta\right)}\right) x_j^{(i)} \\
&= \sum_{i=1}^n \left(y^{(i)}\left(1-f\left({x^{(i)}}^T\beta\right)\right) - \left(1-y^{(i)}\right) f\left({x^{(i)}}^T\beta\right)\right) x_j^{(i)} \\
&= \sum_{i=1}^n \left(y^{(i)} - y^{(i)} f\left({x^{(i)}}^T\beta\right) - f\left({x^{(i)}}^T\beta\right) + y^{(i)} f\left({x^{(i)}}^T\beta\right)\right) x_j^{(i)} \\
&= \sum_{i=1}^n \left(y^{(i)} - f\left({x^{(i)}}^T\beta\right)\right) x_j^{(i)}.
\end{aligned}
$$

Praktisches Parameterschätzung mit dem IWLS Verfahren in **R**

```
lr          = glm(y ~ x[2,], family = 'binomial')          # generalized linear model fit
beta_hat    = lr$coefficients                              # Parametervektorschätzer
```



Anwendungsszenario

Modellformulierung

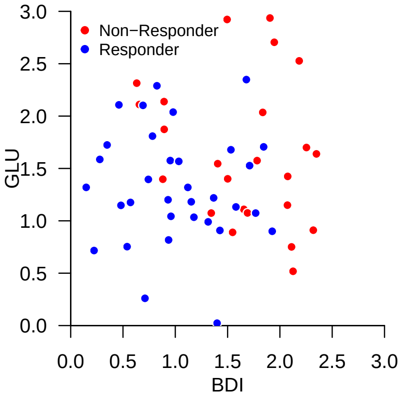
Klassifikation

Lernen

Anwendungsbeispiel

Selbstkontrollfragen

BDI und GLU Werte bei Depressionsdiagnose als Prädiktoren von CBT Erfolg



Anwendungsbeispiel

BDI und GLU Werte bei Depressionsdiagnose als Prädiktoren von CBT Erfolg RES

BDI	GLU	RES
0.74	1.40	1
0.22	0.72	1
0.82	2.29	1
2.07	1.15	0
1.71	1.53	1
1.77	1.07	1
1.95	2.70	0
2.18	2.53	0
0.93	1.20	1
1.34	1.07	0
2.35	1.64	0
1.43	0.91	1
1.66	1.11	0
0.28	1.59	1
2.13	0.52	0
1.37	1.22	1
0.89	2.14	0
0.88	1.40	0
0.98	2.04	1
1.93	0.90	1

LOOCV zur Bestimmung der Featureprädiktivität

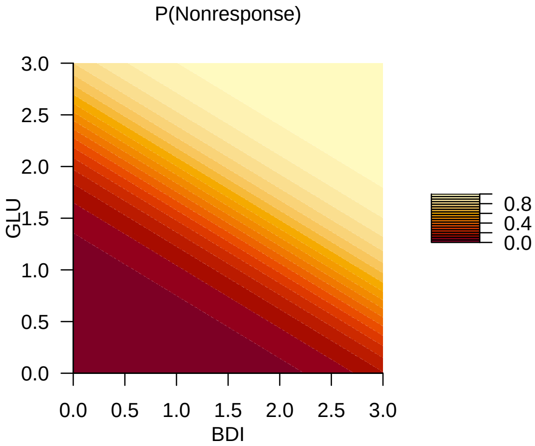
```
D       = read.csv("./12_Daten/12_Logistische_Regression.csv")              # Datensatz
K       = nrow(D)                                                           # Anzahl Cross Folds
p_y     = matrix(rep(NaN, K)  , nrow = 1)                                   # p(y = 1)
y_pred  = matrix(rep(NaN, K*2), nrow = K)                                   # Prädiktionsperformancearray
for(k in 1:K){                                                              # K-fold LOOCV
    x_train      = t(D[-k,1:2])                                             # Trainingsdatensatzfeatures
    y_train      = t(D[-k,3  ])                                             # Trainingsdatensatzlabels
    x_test       = t(D[ k,1:2])                                             # Testdatensatzfeaturevektor
    y_pred[k,1]  = t(D[ k,3])                                               # Testdatensatzfeaturevektorlabel
    n            = ncol(x_train)                                            # n
    m            = nrow(x_train)                                            # m
    lr           = glm(t(y_train) ~ t(x_train), family = 'binomial')        # IWLS Parameterlernen
    beta_hat     = as.matrix(lr$coefficients, nrow = m + 1)                 # Parameterschätzer
    x_test_tilde = rbind(1, x_test)                                         # erweiteter Featurevektor
    p_y[k]       = 1/(1+exp(-t(x_test_tilde) %*% beta_hat))                 # p(y)
    y_pred[k,2]  = as.numeric(p_y[k] >= 0.5)                                # Klassifikationsregel \delta
}
rp      = sum(y_pred[y_pred[,1] == 1,2] == 1)                               # |(1,1)|
rn      = sum(y_pred[y_pred[,1] == 0,2] == 0)                               # |(0,0)|
fp      = sum(y_pred[y_pred[,1] == 0,2] == 1)                               # |(0,1)|
fn      = sum(y_pred[y_pred[,1] == 1,2] == 0)                               # |(1,0)|
ACC     = (rp+rn)/(rp+fp+rn+fn)                                             # Accuracy
SEN     = rp/(rp+fn)                                                        # Sensitivity
SPE     = rn/(rn+fp)                                                        # Specificity
cat("Accuracy   : "  , ACC, ", Sensitivity: " , SEN, ", Specificity: " , SPE)   # Ergebnisausgabe
```

```
Accuracy   :  0.7166667 , Sensitivity:  0.8235294 , Specificity:  0.5769231
```

Evaluation der Non-Response Wahrscheinlichkeit

```
D          = read.csv("./12_Daten/12_Logistische_Regression.csv")             # Datensatz
x          = t(D[,1:2])                                                        # Featurevektoren
y          = t(D[,3])                                                          # Label
n          = ncol(x)                                                           # n
m          = nrow(x)                                                           # m
lr         = glm(t(y_train) ~ t(x_train), family = 'binomial')                 # IWLS Parameterlernen
beta_hat   = as.matrix(lr$coefficients, nrow = m + 1)                          # Parameterschätzer
x_min      = 0                                                                 # GLU/BDI Minimum
x_max      = 3                                                                 # GLU/BDI Maximum
x_res      = 5e2                                                               # GLU/BDI Auflösung
bdi        = seq(x_min, x_max, length.out = x_res)                             # BDI
glu        = seq(x_min, x_max, length.out = x_res)                             # GLU
p_y        = matrix(rep(NaN, x_res*x_res), nrow = x_res)                       # p_{(BDI, GLU}(y=1)
for(i in 1:x_res){                                                             # BDI Iterationen
    for(j in 1:x_res){                                                         # GLU Iterationen
        x_tilde   = rbind(1, bdi[i], glu[j])                                   # \tilde{x}
        p_y[i,j]  = 1/(1+exp(-t(x_tilde) %*% beta_hat))}}                      # p_{(BDI, GLU}(y=1)
```

Evaluation der Non-Response Wahrscheinlichkeit



Anwendungsszenario

Modellformulierung

Klassifikation

Lernen

Anwendungsbeispiel

Selbstkontrollfragen

Selbstkontrollfragen

1. Geben Sie die Definition des Generalisierten Linearen Modells wieder.
2. Geben Sie die Definition der Logistischen Regression als Generalisiertes Lineares Modells wieder.
3. Geben Sie das Theorem zur Mean-Funktion der Logistischen Regression wieder.
4. Geben Sie die Definition des Modells der Logistischen Regression wieder.
5. Erläutern Sie die Generation von Daten unter dem Modell der Logistischen Regression.
6. Geben Sie die Definition der Klassifikationsregel der Logistischen Regression wieder.
7. Erläutern Sie wie mithilfe einer Logistischen Regression die psychotherapeutische Nonresponsewahrscheinlichkeit geschätzt werden kann.

Referenzen

Green, P. J. 1984. "Iteratively Reweighted Least Squares for Maximum Likelihood Estimation, and Some Robust and Resistant Alternatives." *Journal of the Royal Statistical Society: Series B (Methodological)* 46 (2): 149–70. https://doi.org/10.1111/j.2517-6161.1984.tb01288.x.