11.8 Diskriminanzanalyse

Die Diskriminanzanalyse bezieht sich auf dieselbe Erhebungssituation wie die einfaktorielle MANOVA und teilt deren Voraussetzungen (vgl. Abschn. 11.7.1): Beobachtungsobjekte aus p Gruppen liefern Werte auf r quantitativen AVn Y_l. Diese Variablen seien in jeder Gruppe multinormalverteilt mit derselben invertierbaren Kovarianzmatrix, aber u. U. abweichenden Erwartungswertvektoren. Anlass zur Anwendung kann eine zuvor durchgeführte signifikante MANOVA sein, deren unspezifische Alternativhypothese offen lässt, wie genau sich die Gruppenzentroide unterscheiden.

Die Diskriminanzanalyse erzeugt $\min(p-1, r)$ viele Linearkombinationen $LD = b_0 + b_1 Y_1 + \ldots + b_l Y_l + \ldots + b_r Y_r$ der ursprünglichen Variablen, auf denen sich die Gruppenunterschiede im folgenden Sinne besonders deutlich zeigen:[43] Die als *Diskriminanzfunktionen*, oder auch als *Fishers lineare Diskriminanten* bezeichneten LD sind unkorrelierte Variablen, deren jeweiliger F-Bruch aus der einfaktoriellen Varianzanalyse mit der Diskriminanten als AV sukzessive maximal ist. Hier zeigt sich eine Ähnlichkeit zur Hauptkomponentenanalyse (vgl. Abschn. 11.2), die die Gruppenzugehörigkeit der Objekte jedoch nicht berücksichtigt und neue unkorrelierte Variablen mit schrittweise maximaler Varianz bildet.

Die Diskriminanzanalyse lässt sich auch mit der Zielsetzung durchführen, Objekte anhand mehrerer Merkmale möglichst gut hinsichtlich eines bestimmten Kriteriums klassifizieren zu können. Hierfür wird zunächst ein Trainingsdatensatz benötigt, von dessen Objekten sowohl die Werte der diagnostischen Variablen als auch ihre Gruppenzugehörigkeit bekannt sind. Mit diesem Datensatz werden die Koeffizienten der Diskriminanzfunktionen bestimmt, die zur späteren Klassifikation anderer Objekte ohne bekannte Gruppenzugehörigkeit dienen.[44] Die lineare Diskriminanzanalyse lässt sich mit der Funktion `lda()` aus dem `MASS` Paket durchführen.

```
> lda(formula=⟨Modellformel⟩, data=⟨Datensatz⟩, subset=⟨Indexvektor⟩,
+     CV=⟨Kreuzvalidierung⟩, prior=⟨Basiswahrscheinlichkeiten⟩,
+     method="⟨Kovarianzschätzung⟩")
```

Das erste Argument ist eine Modellformel der Form ⟨`UV`⟩ `~` ⟨`AV`⟩. Bei ihr ist abweichend von den bisher betrachteten linearen Modellen die links von der `~` stehende, vorherzusagende Variable ein Faktor der Gruppenzugehörigkeiten, während die quantitativen AVn die Rolle der Prädiktoren auf der rechten Seite einnehmen. Stammen die in der Modellformel verwendeten Variablen aus einem Datensatz, ist dieser unter `data` zu nennen. Das Argument `subset` erlaubt es, nur eine Auswahl der Fälle einfließen zu lassen, etwa wenn eine Trainingsstichprobe als zufällige Teilmenge eines Datensatzes gebildet werden soll. `subset` erwartet einen Indexvektor, der sich auf die Zeilen des Datensatzes bezieht.

In der Voreinstellung verwendet `lda()` die relativen Häufigkeiten der Gruppen als Maß für ihre Auftretenswahrscheinlichkeiten in der Population. Über das Argument `prior` lassen sich letztere auch explizit in Form eines Vektors in der Reihenfolge der Stufen von ⟨`UV`⟩ vorgeben. Setzt man

[43] Lässt man auch quadratische Funktionen der ursprünglichen Variablen zu, ergibt sich die quadratische Diskriminanzanalyse. Sie wird mit der Funktion `qda()` aus dem `MASS` Paket berechnet.

[44] Für weitere Klassifikationsverfahren wie Varianten der Clusteranalyse, CART-Modelle oder support vector machines vgl. die Abschnitte `Cluster Analysis`, `Multivariate Statistics` und `Machine Learning & Statistical Learning` der Task Views (R Development Core Team, 2011a). Auch die logistische Regression (vgl. Abschn. 8.1) lässt sich für eine dichotome Klassifikation verwenden.

das Argument `method="mve"`, verwendet `lda()` eine robuste Schätzung von Mittelwerten und Kovarianzmatrix.

Das Beispiel verwendet dieselben Daten wie die einfaktorielle MANOVA (vgl. Abschn. 11.7.1), wobei die ungleichen Gruppenhäufigkeiten zunächst als Indikator für ihre Wahrscheinlichkeiten dienen sollen.

```
> Ydf1 <- data.frame(IVman, DV1=Ym1[ , 1], DV2=Ym1[ , 2])
> library(MASS)                                          # für lda()
> (ldaRes <- lda(IVman ~ DV1 + DV2, data=Ydf1))
Call:
lda(IVman ~ DV1 + DV2, data = Ydf1)

Prior probabilities of groups:
        1         2         3
0.2500000 0.4166667 0.3333333

Group means:
        DV1    DV2
1 -3.266667   5.60
2  2.360000   4.04
3 -0.350000  -0.40

Coefficients of linear discriminants:
            LD1          LD2
DV1  0.06710397 -0.27380306
DV2 -0.31861364 -0.07850291

Proportion of trace:
   LD1     LD2
0.6327  0.3673
```

Die Ausgabe nennt unter der Überschrift `Prior probabilities of groups` die angenommenen Gruppenwahrscheinlichkeiten. Unter `Group means` folgt eine zeilenweise aus den Vektoren der Gruppenzentroide zusammengestellte Matrix, die in der von `lda()` zurückgegebenen Liste in der Komponente `means` enthalten ist. Die Koeffizienten b_l der Diskriminanzfunktionen finden sich spaltenweise unter `Coefficients of linear discriminants`, das Ergebnis speichert diese Matrix in der Komponente `scaling`. In der Ausgabe fehlen die für eine Interpretation der Ergebnisse meist nicht interessanten absoluten Terme b_0 der Linearkombinationen. Unter `Proportion of trace` lässt sich der Anteil der von jeder Diskriminanzfunktion aufgeklärten Varianz an der Gesamtvarianz zwischen den Gruppen im unten näher erläuterten Sinn ablesen.

Das Argument `CV=TRUE` (Voreinstellung ist `FALSE`) bewirkt eine Kreuzvalidierung, wobei gleichzeitig für jede Beobachtung und jede Gruppe die a-posteriori Wahrscheinlichkeit i. S. von Bayes berechnet wird, dass die Beobachtung zu einer Gruppe gehört. Die Matrix dieser Wahrscheinlichkeiten findet sich dann in der Komponente `posterior` der ausgegebenen Liste. Dagegen unterbleibt in diesem Fall die Berechnung der Koeffizienten für die Diskriminanzfunktionen.

```
> ldaP <- lda(IVman ~ DV1 + DV2, CV=TRUE, data=Ydf1)
```

```
> ldaP$posterior                    # Wahrscheinlichkeiten ...
```

Die aus der Regression bekannte `predict(⟨lda-Objekt⟩, ⟨Datensatz⟩)` Funktion (vgl. Abschn. 6.4) dient zur Vorhersage der Gruppenzugehörigkeiten auf Basis eines von `lda()` ausgegebenen Objekts. Dazu ist als zweites Argument ein Datensatz mit Variablen zu nennen, die dieselben Namen wie die AVn aus der ursprünglichen Analyse tragen. Die von `predict()` ausgegebene Liste enthält in der Komponente `x` die Diskriminanten selbst und in der Komponente `class` die vorhergesagte Kategorie. Die Güte der Vorhersage lässt sich für die Trainingsstichprobe etwa an der Konfusionsmatrix gemeinsamer Häufigkeiten von tatsächlichen und vorhergesagten Gruppenzugehörigkeiten ablesen (für weitere Maße der Übereinstimmung kategorialer Variablen vgl. Abschn. 9.2.6, 9.3.3).

```
> ldaPred <- predict(ldaRes, Ydf1)  # Vorhersage für ursprüngliche Daten
> head(ldaPred$x)                   # Diskriminanten
         LD1          LD2
1  0.4166295   0.98818001
2 -2.6352990  -0.34445522
3 -2.7193092   1.37686604
4 -1.0591370   1.49557754
5 -3.7253478  -0.03235784
6  0.8694511   0.51907680

> head(ldaPred$class)               # Klassifikation
[1] 3 1 1 1 1 3
Levels: 1 2 3

# Kontingenztafel tatsächlicher und vorhergesagter Kategorien
> cTab <- table(IVman, ldaPred$class, dnn=c("IVman", "ldaPred"))
> addmargins(cTab)
          ldaPred
IVman   1   2   3  Sum
    1   9   3   3   15
    2   3  19   3   25
    3   1   6  13   20
  Sum  13  28  19   60

> sum(diag(cTab)) / sum(cTab)       # prozentuale Übereinstimmung
[1] 0.6833333
```

Die manuelle Kontrolle beruht auf den in Abschn. 11.9.9 berechneten Matrizen $\boldsymbol{B}$ (SSP-Matrix der durch das zugehörige Gruppenzentroid ersetzten Daten) und $\boldsymbol{W}$ (SSP-Matrix der Residuen). Die Koeffizientenvektoren der Diskriminanzfunktionen erhält man aus den Eigenvektoren von $\boldsymbol{W}^{-1}\boldsymbol{B}$. Diese werden dafür zum einen so umskaliert, dass die Residual-Quadratsumme der univariaten Varianzanalysen mit je einer Diskriminante als AV gleich $N - p$, die mittlere Residual-Quadratsumme also gleich 1 ist. Die F-Brüche dieser Varianzanalysen sind gleich den Eigenwerten von $\boldsymbol{W}^{-1}\boldsymbol{B}$, die mit dem Quotient der Freiheitsgrade der Quadratsummen innerhalb $(N - p)$ und zwischen den Gruppen $(p - 1)$ multipliziert wurden. Zum anderen

werden die Diskriminanten so verschoben, dass ihr Mittelwert jeweils 0 beträgt. Der Anteil der Eigenwerte von $\boldsymbol{W}^{-1}\boldsymbol{B}$ an ihrer Summe, also an der Spur von $\boldsymbol{W}^{-1}\boldsymbol{B}$, wird von `lda()` unter `Proportion of trace` genannt.

```
> eigWinvB <- eigen(solve(WW) %*% BB)   # Eigenwerte, -vektoren W^-1 * B
> eigVec   <- eigWinvB$vectors          # Eigenvektoren
> eigVal   <- eigWinvB$values           # Eigenwerte
> p        <- nlevels(IVman)            # Anzahl Gruppen
> N        <- sum(Nj)                   # gesamt-N
> My       <- colMeans(Ym1)             # Mittelwerte Variablen

# Proportion of trace, alternativ: eigVal / sum(eigVal)
> eigVal / sum(diag(solve(WW) %*% BB))
[1] 0.6327184 0.3672816

# Skalierungsfaktoren für Eigenvektoren
> scl <- sqrt((N-p) / diag(t(eigVec) %*% WW %*% eigVec))
> b0  <- -scl * t(eigVec) %*% My        # absolute Terme b_0

# Skalierung der Eigenvektoren -> Matrix mit Koeffizienten b_k
> (bk <- eigVec %*% diag(scl))
            [,1]         [,2]
[1,]  0.06710397  -0.27380306
[2,] -0.31861364  -0.07850291

# prüfe, ob Diskriminanten mit Ergebnis von predict() übereinstimmen
> ld <- sweep(Ym1 %*% bk, 2, b0, "+")   # Diskriminanten
> all.equal(ld, ldaPred$x, check.attributes=FALSE)
[1] TRUE

# univariate ANOVAs mit je einer Diskriminante als AV: SSw=N-p, MSw=1
> anova(lm(ld[ , 1] ~ IVman))           # ANOVA mit 1. Diskriminante
Analysis of Variance Table
Response: ld1
           Df  Sum Sq  Mean Sq  F value     Pr(>F)
IVman       2  39.651   19.826   19.826  2.911e-07 ***
Residuals  57  57.000    1.000

> anova(lm(ld[ , 2] ~ IVman))           # ANOVA mit 2. Diskriminante
Analysis of Variance Table
Response: ld2
           Df  Sum Sq  Mean Sq  F value     Pr(>F)
IVman       2  23.017   11.508   11.508  6.335e-05 ***
Residuals  57  57.000   1.0000

# F-Werte der ANOVAs aus Eigenwerten von W^-1 * B
> ((N-p) / (p-1)) * eigVal
```

```
[1] 19.82570 11.50846
```

Wurden der Diskriminanzanalyse gleiche Gruppenwahrscheinlichkeiten zugrunde gelegt, ergibt sich die vorhergesagte Gruppenzugehörigkeit für eine Beobachtung aus dem minimalen euklidischen Abstand zu den Gruppenzentroiden im durch die Diskriminanzfunktionen gebildeten Koordinatensystem: Dazu sind die Diskriminanten für alle Beobachtungen zu berechnen und die Gruppenmittelwerte der Trainingsstichprobe auf jeder Diskriminante zu bilden. Für die zu klassifizierende Beobachtung wird jene Gruppe als Vorhersage ausgewählt, deren Zentroid am nächsten an der Beobachtung liegt.

```
# Diskriminanzanalyse mit Annahme gleicher Gruppenwahrscheinlichkeiten
> priorP <- rep(1/nlevels(IVman), nlevels(IVman))       # gleiche Wkt.
> ldaEq  <- lda(IVman ~ DV1 + DV2, prior=priorP, data=Ydf1)
> predEq <- predict(ldaEq, Ydf1)              # Diskrim., Vorhersage
> LDmat  <- predEq$x                          # Diskriminanten

# Datensatz der Gruppenzentroide
> ctrDf <- aggregate(cbind(LD1, LD2) ~ IVman, FUN=mean, data=LDmat)
> ctrLD <- data.matrix(ctrDf[ , -1])          # Matrix Gruppenzentroide

# Matrizen der Differenzvektoren der Beobachtungen zu jedem Zentroid
> diffMat1 <- sweep(LDmat, 2, ctrLD[1, ], "-")        # zu Zentroid 1
> diffMat2 <- sweep(LDmat, 2, ctrLD[2, ], "-")        # zu Zentroid 2
> diffMat3 <- sweep(LDmat, 2, ctrLD[3, ], "-")        # zu Zentroid 3

# euklidische Distanzen als jeweilige Länge des Differenzvektors
> dst1   <- sqrt(diag(tcrossprod(diffMat1)))          # zu Zentroid 1
> dst2   <- sqrt(diag(tcrossprod(diffMat2)))          # zu Zentroid 2
> dst3   <- sqrt(diag(tcrossprod(diffMat3)))          # zu Zentroid 3
> dstMat <- cbind(dst1, dst2, dst3)           # Matrix: alle Distanzen

# jede Zeile: identifiziere Spalte mit minimaler Distanz -> Vorhersage
> dstPred <- apply(dstMat, 1, which.min)
> head(dstPred)                               # Klassifikation
1 2 3 4 5 6
3 1 1 1 1 3

# prüfe auf Übereinstimmung mit Ergebnis von predict()
> all(dstPred == unclass(predEq$class))
[1] TRUE
```