Diskriminanzanalyse

Schmidt, Sophie C.

21 August, 2020

# Diskriminanzanalyse

Die Diskriminanzanalyse ist jetzt schon etwas, das manche Leute als “Machine Learning” bezeichnen.

Faaaancy.

Oder zumindest klingt es so. Aber ein Witz unter Statistikern geht ungefähr so “Steht Machine Learning drauf, ist aber Lineare Regression drin”. Und das bei vielen anderen auch: “Steht ANOVA drauf, ist aber lineare Regression drin”…

<https://media.makeameme.org/created/when-you-advertise-f81897f53a.jpg>

<https://media.makeameme.org/created/when-you-advertise-f81897f53a.jpg>

Die LDA entwickelt eine Schätzfunktion anhand der sie die Daten mithilfe von metrischen Variablen in Gruppen einteilt.

In dieser Aufgabe untersuchen wir, ob sich das Geschlecht der Piraten anhand aller metrischer Variablen in dem Datensatz vorhersagen lässt. Dafür reduzieren wir als erstes unseren Datensatz auf die benutzten Variablen:

library(yarrr)  
data("pirates")  
  
pir <- pirates[,c(2:5,8:10,14:15,17)]

In dem wir einen Trainingsdatensatz und einen Test-Datensatz benutzen, können wir schauen, ob die berechnete Diskriminanzfunktion eine ordentliche Vorhersage unserer Gruppen macht.

Also splitten wir den Piratendatensatz erst einmal in zwei Gruppen:

train <- sample(nrow(pir), size = 0.75\*nrow(pir)) # so erstelle ich einen Vektor aus zufällig gewählte Zahlen von 0 bis der Anzahl von Datensätzen im Piratendatensatz in der Größe von von 75% aus der Anzahl von Datensätzen im Piratendatensatz. Ich bekomme also einen Vektor namens "train" mit 750 Zahlen, die zufällig aus dem Bereich von 0 bis 1000 ausgewählt wurden.   
  
train\_pir.df <- as.data.frame(pir[train, ]) # hiermit wähle ich mithilfe der Zufallswerte zufällige Zeilen aus dem Piratendatensatz aus  
  
test\_pir.df <- as.data.frame(pir[-train, ]) # hiermit nehme ich die Zeilen aus dem Datensatz, die nicht mit dem Zufallsvektor angesprochen werden (-train)  
  
## train\_pir.df + test\_pir.df = pirates

Jetzt müssen wir wieder die Daten normalisieren, genau wie bei der PCA. In der lda()-Funktion, die wir für die lineare Diskriminanzanalyse nutzen, gibt es dafür leider keine extra Argumente. Deswegen müssen wir das im Vorhinein machen. Dazu nutzen wir das Paket caret und die Funktion preprocess mit den Argumenten method = c("center", "scale") (erinnert euch: das sind die gleichen Argumente wie beider pca).:

library(caret)  
library(tidyr)  
  
# die Parameter werden geschätzt  
preproc.param <- train\_pir.df %>%   
 preProcess(method = c("center", "scale"))  
  
# anhand der geschätzten Parameter werden die Daten transfomiert (Funktion "predict")  
# 1. der Trainingsdatensatz  
train.transformed <- preproc.param %>%   
 predict(train\_pir.df)   
  
# 2. der Testdatensatz  
test.transformed <- preproc.param %>%   
 predict(test\_pir.df)

Jetzt können wir die Diskrimanzanalyse mit den transformierten Daten beginnen. Dafür benötigen wir das Paket MASS. Die Modellberechnung folgt dabei der gleichen Logik wie bei der linearen Regression.

Wir möchten jetzt einmal alle metrischen Variablen nutzen, um vorherzusagen, welches Geschlecht (Spalte sex) unsere Piraten haben. Alle anderen Parameter im Datensatz können wir einfach mit “.” ansprechen:

library(MASS)  
model <- lda(sex~., data = train.transformed)

Das Modell können wir uns einmal ansehen:

model

Oh. Eine Menge Informationen. Was will uns R damit sagen?

Zuerst, wie immer, welche Funktion wir aufgerufen haben.

* “Prior probabilities of groups”, das sind ganz einfach die Verhältnisse im Trainingsdatensatz. 46,7% sind Frauen, 49,2% sind Männer und 4,1% sind “other”.
* “Group means” sind die Mittelwerte jeder Variablen in den einzelnen Gruppen. Da die Daten normalisiert wurden, schwanken sie häufig um 0!
* “Coefficients of linear disciminants” sind die Faktoren, die vor die jeweilige Variable gesetzt werden, um die lineare Funktion zu bestimmen. z. B. für LD1 gilt $ LD1 = -0,21*age + 0,29*height + 0,04*weight - 0,42*  tattoos + 0,01 \* tchests + 0,02 \* parrots + 0,008 \* sword.time + 2,4 \* beard.length + 0,01 \* grog $
* “Proportion of trace” (trace = “Spur” im deutschen) beschreibt den Anteil der Varianz zwischen den Klassen, der durch LD1 und LD2 erklärt wird. Wie bei der MANOVA, nur wird hier kein Signifikanztest gemacht

Das Modell zu plotten, erklärt eine Menge. Ganz einfach mit der base - Funktion plot() :

plot(model)

Hier sehn wir das “neue Koordinatensystem”, das auf LD1 und LD2 basiert und die einzelnen Punkte, beschriftet mit ihrer Gruppenzugehörigkeit. Und wie man sieht, trennen sich die Gruppen entlang der Achse LD1 - weswegen dort die “proportion of trace” sehr hoch ist. Die Unterschiede zwischen den Gruppen werden durch LD1 erklärt.

Jetzt gilt es, mit dem Testdatensatz zu schauen, ob die berechnete Funktion die Gruppenzugehörigkeit (female, male, other) ordentlich vorhersagt. Spannung! Wie gut ist unser Modell?

Zuerst berechnen wir die Vorhersage mit der Funktion predict():

# Vorhersage berechnen  
predictions <- model %>% # anhand des modells, berechne  
 predict(test.transformed) # die Vorhersage mit dem Testdatensatz

Die Modell-Genauigkeit berechnen wir mit dem Mittelwert (1 für TRUE, 0 für FALSE), der Frage: Wie häufig stimmt in predictions das vorhergesagt Geschlecht (predictions$class) mit dem tatsächlichen Geschlecht (test.transformed$sex)) überein?

mean(predictions$class==test.transformed$sex) #immer wenn die Aussage stimmt: TRUE = 1, wenn nicht FALSE = 0.

Der Mittelwert ist fast 1. Sehr gut! Fast immer ist das Geschlecht richtig vorhergesagt worden!

Was kann ich noch mit den vorhergesagten Datensatz machen?

In “predictions” stecken noch weitere Informationen. Es ist vom Datentyp eine “Liste”, was bedeutet, dass verschiedene Datensätze darin vereinigt werden, die unterschiedliche Datentypen haben. Unter “class” findet sich ein Vektor mit der bereits abgefragten Vorhersage, in welcher Gruppe der Pirat gehört. Unter “posterior” liegt eine Matrix, die als Spalte die Gruppen (female, male, other) und als Zeilen die Individuen (Piraten) hat. Die Werte sind die Wahrscheinlichkeit, dass der Pirat in diese Gruppe gehört. Die Entscheidung, dass ein Individuum ein bestimmtes Geschlecht hat, wird in der Regel dann gefällt, wenn an dieser Stelle die Wahrscheinlichkeit für ein Geschlecht über 0.5 liegt.

Unter “x” findet sich die Matrix mit LD1 und LD2 pro Individuum.

Schaut es euch an!

# vorhergesagte Gruppenzugehörigkeit  
head(predictions$class, 6) # gib die ersten 6 Werte des Vektors  
  
# Vorhergesagte Wahrscheinlichkeit, in welche Gruppen der Pirat gehört  
head(predictions$posterior, 6) # zeig die erstene 6 Zeilen (Zeilennamen = ID)  
  
# LD1 und LD2  
head(predictions$x, 3) # zeig die erstene 3 Zeilen (Zeilennamen = ID)

Diese Daten können wir jetzt auch nutzen, um sie schön in ggplot darzustellen. Zuerst verbinden wir den ursprünglichen normalisierten Datensatz mit LD1 und LD2 und haben dann einen neuen Datensatz, mit dem wir ggplot nutzen können:

lda.data <- cbind(train.transformed, predict(model)$x) # cbind = column bind = verbinde die Spalten  
  
ggplot(lda.data, aes(LD1, LD2)) +  
 geom\_point(aes(color = sex)) +  
 theme\_bw()+  
 ggtitle("Diskriminanzanalyse nach Geschlecht")

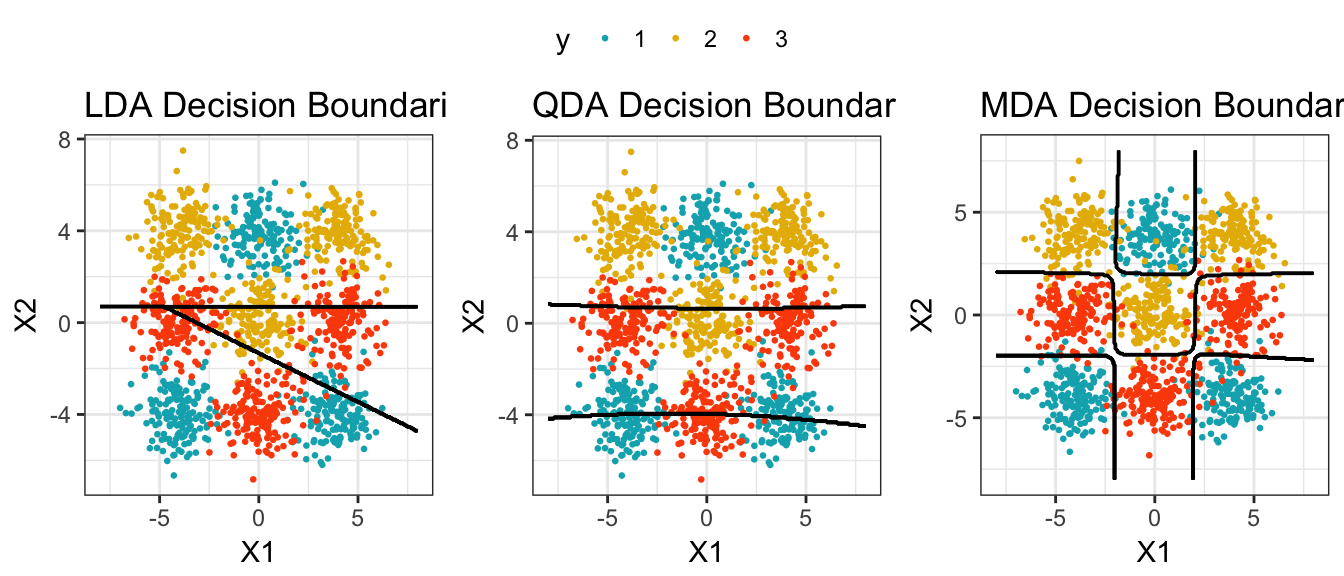
Sehr schön!

Es gibt nicht nur lineare Diskriminanzanalysen, sondern auch quadratische, gemischte, regularisierte und flexible Diskriminanzanalyse. Die quadratische und die gemischt thematisieren wir hier noch knapp.

## Weitere Diskriminanzanalysen

Die quadratische Diskriminanzanalyse ist für große Datensätze geeignet und setzt keine Varianzgleichheit in den verschiedenen Gruppen voraus.

Bei der LDA wird davon ausgegangen, dass jede Gruppe aus einer einzigen Normalverteilung stammt. In der “Gemischten” Diskriminanzanalyse (Mixture Discriminance Analysis - MDA) wird davon ausgegangen, dass die Gruppen aus einer Mischung aus Subgruppen besteht, die jeweils etwas andere Charakteristika aufweisen. Die MDA ist dann deutlich besser, wenn die Untergruppen “gemischt” auftreten, wie in diesem Bild:



Für die MDA benötigt man ein neues R-Paket namens mda, in dem die Funktion mda() zu finden ist.

Die flexible Diskriinanzanalys (FDA) benutzt nicht-lineare Kombinationen der vorhersagenden Variablen, wie zB splines (“glättende Kurven”). Das ist dann geeignet, wenn die Daten nicht normal sind oder keine linearen Zusammenhänge zwischen den Variablen bestehen. Die Funktion fda in dem Paket mda setzt sie um.

Die regularisierte Diskriminanzanalyse ist bei Multikollinearität eine gute Wahl, da es für die unterschiedlichen Gruppen von unterschiedlichen Kovarianzmatrizen ausgeht. Damit stellt sie einen mittelweg zwischen der LDA und der QDA dar. In R benutzt man die Funktion rda in dem Paket klaR dafür.

Die beschriebenen Funktionen “funktionieren” in R alle gleich und fast genauso wie die LDA. Der Unterschied ist, dass man sie nicht mehr gut grafisch darstellen kann. Als Beispiel berechnen wir nur noch eine quadratische Diskriminanzanalyse:

## Quadratische Diskriminanzanalyse

Die Quadratische Diskriminanzanalyse (QDA) ist etwas flexibler als die LDA, da sie nicht die Varianzgleichheit in den verschiedenen Gruppen voraussetzt. LDA funktioniert besser, wenn man einen kleinen Trainingsdatensatz hat. Da unser Piratendatensatz sehr groß ist, wird für ihn eigentlich QDA empfohlen.

Zuerst die Modellberechnung:

model\_qda <- qda(sex~., data = train.transformed)  
model\_qda

Wie man sieht, sind die angezeigten Werte gleich (die Mittelwerte bleiben ja gleich). Es fehlen aber LD1 und LD2. Das liegt daran, dass bei einer QDA keine LINEAREN Berechnungen mehr durchgeführt werden. Aus diesem Grund können die Daten auch schlecht in den zwei-dimensionalen Raum geplottet werden. Wir habe also nur die Möglichkeit, die Berechnungen durchzuführen und uns die Ergebnisse mathematisch einzuschätzen:

# Vorhersage berechnen  
predictions\_qda <- model\_qda %>%   
 predict(test.transformed)  
  
# Model accuracy  
mean(predictions$class == test.transformed$sex)

Oha. Das Ergebnis ist ein klein wenig schlechter als bei der LDA. Das liegt daran, dass eine lineare Funktion einfach besser auf die Daten passt als eine quadratische.

## FDA

Nur schnell der Code, falls ihr das später einmal braucht:

library(mda)  
# Fit the model  
model\_fda <- fda(sex~., data = train.transformed)  
# Make predictions  
predicted.classes\_fda <- model\_fda %>% predict(test.transformed)  
# Model accuracy  
mean(predicted.classes\_fda == test.transformed$sex)

## MDA

Nur schnell der Code, falls ihr das später einmal braucht:

library(klaR)  
# Fit the model  
model\_rda <- rda(sex~., data = train.transformed)  
# Make predictions  
predictions\_rda <- model\_rda %>% predict(test.transformed)  
# Model accuracy  
mean(predictions\_rda$class == test.transformed$sex)

Man sieht bei FDA und MDA werden die Vorhersagen nicht besser als bei der LDA.

Deswegen:

**Aufgabe** Berechnet eine LDA für die Vorhersagbarkeit der Pinguinspezies anhand von Flügelmaß, der zwei Schnabelmaße und Gewicht. Bereinigt den Datensatz von NA - Werten und benutzt einen Trainingsdatensatz, in dem 70% der Daten enthalten sind.

## Zusammenfassung:

Die Diskriminanzanalyse benötigt einen Trainingsdatensatz, mit dem ein Algorithmus berechnet wird, der unsere Gruppen trennen soll. Mit einem Testdatensatz können wir dann testen, ob die Berechnung gut ist. Wenn sie das ist, können wir automatisch weitere neue Datenpunkte mit der berechneten Funktion klassifizieren.

Das haben Sie alles hier kennenglernt. Glückwunsch!