Diskriminanzanalyse und Klassifikation

Vitaly Belik

Institut für Veterinär-Epidemiologie und Biometrie, FU Berlin

17/12/2019

Wiederholung

Was ist der Unterschied zwischen linearen *Diskriminanzanalyse* (als eine Art Klassifikationsverfahren) und *Clustering*?

LDA ist eng mit der Varianzanalyse (ANOVA) und der Regressionsanalyse verbunden, die auch versuchen, eine abhängige Variable als lineare Kombination anderer Merkmale oder Messungen auszudrücken. ANOVA verwendet jedoch qualitative erklärende Variablen und eine kontinuierliche abhängige Variable, während eine Diskriminanzanalyse kontinuierliche erklärende Variablen und eine qualitative abhängige Variable (dh Klassenbezeichnung) enthält.

Die logistische Regression und die Probit-Regression sind der LDA ähnlicher als die Varianzanalyse, da sie die qualitative Variable auch anhand kontinuierlicher erklärender Variablen erklären. Diese anderen Methoden sind in Anwendungen vorzuziehen, bei denen kein Grund zu der Annahme besteht, dass die unabhängigen Variablen *normal verteilt* sind (obwohl es auch für die leichte Abweichung davon auch funktioniert). Dies ist die Grundannahme der LDA-Methode.

Diskriminanzanalyse

Diskriminanzanalyse

- Diskiminanzanalyse benutzt man um die Wahrscheinlichkeit zu bestimmen, dass eine Beobachtung zu einer *bestimmten Klasse* oder Kategorie aus mehreren zu gehören.
- Die Prädiktoren können sowohl kontinuierlich als auch kategoriell sein

Lineare Diskriminanzanalyse

- Der LDA-Algorithmus beginnt mit der Suche nach Richtungen, die die Trennung zwischen Klassen maximieren, und verwendet diese Richtungen, um die Klassen vorherzusagen.
- Diese Richtungen, lineare Diskriminanten genannt, sind eine lineare Kombination von Prädiktorvariablen.
- Annahmen:
  1) Normalverteilung der Prädiktoren
  2) Die Klassen haben die gleichen Varianzen oder Kovarianzmatrizen.

Funktionsweise

Es wird nach einer linearen Kombination (2-dimensionaler Fall) $w_x x + w_z z$ gesucht dass das Verhältnis maximiert:

$$\frac{SS_{\text{between}}}{SS_{\text{within}}}$$

Lineare Diskriminanzanalyse: Vorarbeit

- Überprüfen Sie die univariaten Verteilungen jeder Variablen und stellen Sie sicher, dass sie normal verteilt sind. Andernfalls können Sie sie mit log und Wurzel für Exponentialverteilungen und mit Box-Cox für schräge Verteilungen transformieren.
- Entfernen Sie Ausreißer aus Ihren Daten und standardisieren Sie die Variablen, um deren Maßstab vergleichbar zu machen.

Box-Cox-Transformationen

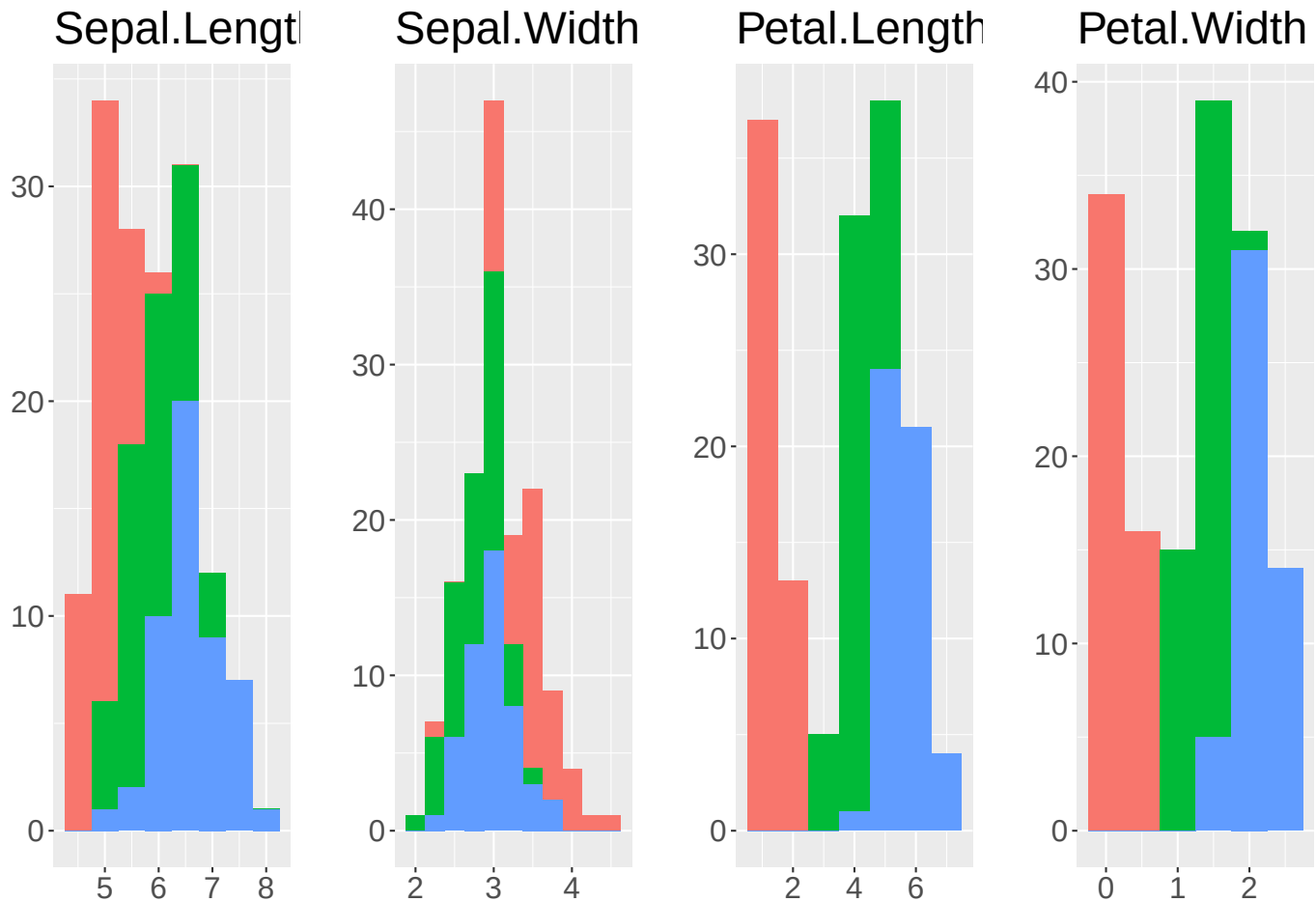
$$y(\lambda) = \begin{cases} \dfrac{y^\lambda - 1}{\log y} & \text{if } \lambda \neq 0 \\ \log y & \text{if } \lambda = 0 \end{cases}$$

Fisher's Iris-Datensatz

Figure 1: R. A. Fisher (1936). The use of multiple measurements in taxonomic problems. Annals of Eugenics. 7 (2): 179–188

```
library(tidyverse)
library(caret)
data("iris")
knitr::kable(head(iris))
```

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa


Sepal.Lengt
Sepal.Width
Petal.Length
Petal.Width
30
20
10
0
40
30
20
10
0
30
20
10
0
40
30
20
10
0
5
6
7
8
2
3
4
2
4
6
0
1
2

Lineare Diskriminanzanalyse(1)

Der Datensatz wird in Test-Datensatz und Training-Datensatz aufgeteilt.

```
set.seed(123)
training.samples <- iris$Species %>% createDataPartition(p = 0.8, list = FALSE)
train.data <- iris[training.samples, ]
test.data <- iris[-training.samples, ]
```

Die Daten werden normalisiert. In diesem Fall, können die Gewichte von Diskriminanten als Maß für die Wichtigkeit der Merkmale verwendet werden.

```
preproc.param <- train.data %>% preProcess(method = c("center", "scale"))
train.transformed <- preproc.param %>% predict(train.data)
test.transformed <- preproc.param %>% predict(test.data)
```

Lineare Diskriminanzanalyse(3)

```
library(MASS)
model <- lda(Species ~ ., data = train.transformed)
predictions <- model %>% predict(test.transformed)
mean(predictions$class == test.transformed$Species)
```

```
## [1] 0.9666667
```

```
model
```

```
## Call:
## lda(Species ~ ., data = train.transformed)
##
## Prior probabilities of groups:
##     setosa versicolor  virginica
##  0.3333333  0.3333333  0.3333333
##
## Group means:
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa       -1.0112835  0.78048647   -1.2900001  -1.2453195
## versicolor    0.1014181 -0.68674658    0.2566029   0.1472614
## virginica     0.9098654 -0.09373989    1.0333972   1.0980581
##
## Coefficients of linear discriminants:
##                     LD1         LD2
## Sepal.Length  0.6794973  0.04463786
## Sepal.Width   0.6565085 -1.00330120
## Petal.Length -3.8365047  1.44176147
## Petal.Width  -2.2722313 -1.96516251
##
## Proportion of trace:
##    LD1    LD2
## 0.9902 0.0098
```

Visualisierung von Ergebnissen I

```
predictions <- model %>% predict(test.transformed)
head(predictions)
```

```
## $class
##  [1] setosa     setosa     setosa     setosa     setosa     setosa
##  [7] setosa     setosa     setosa     setosa     versicolor versicolor
## [13] versicolor versicolor versicolor versicolor versicolor versicolor
## [19] versicolor versicolor virginica  virginica  virginica  virginica
## [25] virginica  virginica  virginica  versicolor virginica  virginica
## Levels: setosa versicolor virginica
##
## $posterior
##           setosa   versicolor    virginica
## 1   1.000000e+00 3.978425e-22 1.319337e-43
## 2   1.000000e+00 1.038098e-17 3.967605e-38
## 6   1.000000e+00 2.882148e-21 2.041612e-41
## 16  1.000000e+00 8.381782e-28 7.486309e-50
## 23  1.000000e+00 6.531615e-25 3.414098e-47
## 34  1.000000e+00 1.089899e-28 7.865614e-52
## 35  1.000000e+00 1.449641e-17 8.617776e-38
## 38  1.000000e+00 2.569637e-23 2.490647e-45
## 44  1.000000e+00 9.532279e-16 1.078894e-33
## 47  1.000000e+00 1.357831e-22 7.647431e-44
## 51  3.896613e-18 9.999518e-01 4.821738e-05
## 60  2.156329e-20 9.997875e-01 2.124713e-04
## 64  2.984097e-23 9.970966e-01 2.903446e-03
## 73  1.426307e-28 8.761315e-01 1.238685e-01
## 74  7.061573e-22 9.998138e-01 1.862221e-04
## 87  6.615078e-21 9.991080e-01 8.919736e-04
## 91  1.396073e-22 9.997365e-01 2.634948e-04
## 94  2.215203e-14 1.000000e+00 2.929967e-08
## 95  9.053615e-21 9.998731e-01 1.268678e-04
## 97  6.118802e-19 9.999554e-01 4.460634e-05
## 104 5.684420e-38 1.055968e-03 9.989440e-01
## 109 2.940937e-42 2.018347e-04 9.997982e-01
## 111 1.464269e-31 1.419173e-02 9.858083e-01
```

Visualisierung von Ergebnissen II

```
## 113 1.023351e-38 1.717615e-04 9.998282e-01
## 116 9.953348e-40 1.990495e-05 9.999801e-01
## 120 3.529426e-33 2.862129e-01 7.137871e-01
## 127 1.757931e-29 2.351857e-01 7.648143e-01
## 134 3.846340e-28 8.063778e-01 1.936222e-01
## 138 1.920885e-34 6.617982e-03 9.933820e-01
## 147 7.176867e-36 6.395225e-03 9.936048e-01
##
## $x
##            LD1         LD2
## 1    8.1629386 -0.50527678
## 2    7.2027133  0.71110616
## 6    7.8162430 -1.73271506
## 16   9.2840285 -3.10506837
## 23   8.7872943 -1.09878034
## 34   9.6017637 -2.20200431
## 35   7.1488541  0.54633214
## 38   8.4569312 -0.50932474
## 44   6.4915104 -1.35588952
## 47   8.2157619 -1.08017093
## 51  -1.3891224 -0.03180619
## 60  -1.8942686  0.45721896
## 64  -2.5795707  0.65766782
## 73  -3.7839909  1.55889831
## 74  -2.1581786  1.40630549
## 87  -2.0769055 -0.05362730
## 91  -2.3086075  1.62423854
## 94  -0.2436776  1.95140633
## 95  -1.9314725  0.97127628
## 97  -1.5316307  0.48575787
## 104 -5.5155322  0.38658118
## 109 -6.2714692  1.54980508
## 111 -4.3863191 -1.24526156
## 113 -5.6196997 -0.66840529
## 116 -5.7667803 -1.84422852
## 120 -4.7162917  2.35998730
```

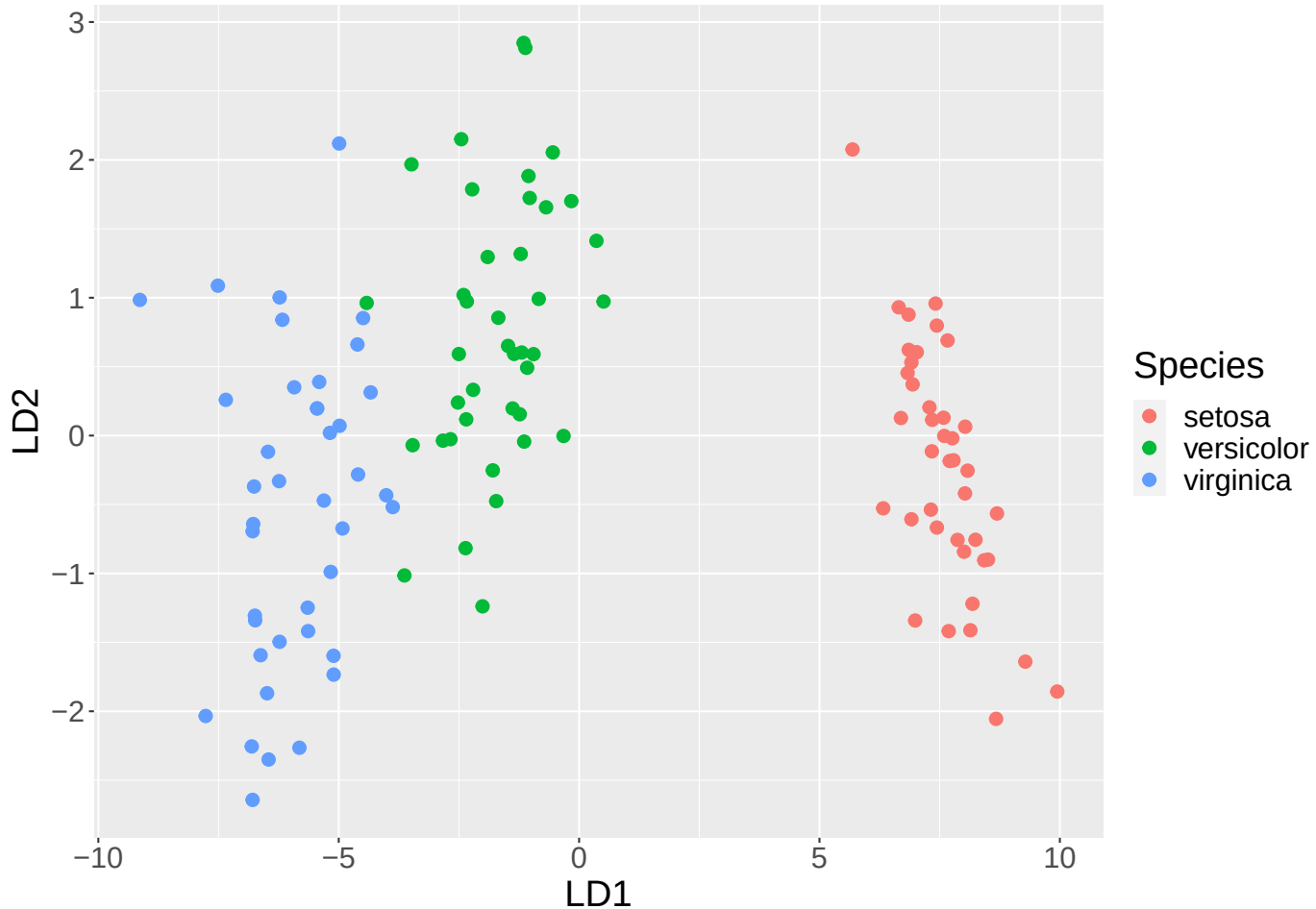
Visualisierung von Ergebnissen III

```
## 127 -4.0396234 -0.01782327
## 134 -3.7311676  0.98400416
## 138 -4.9012893 -0.17950933
## 147 -5.1622599  0.63288729
```

```
# plot(model)
lda.data <- cbind(train.transformed, predict(model)$x)
```

Visualisierung von Ergebnissen

```
# plot(model)
lda.data <- cbind(train.transformed, predict(model)$x)
ggplot(lda.data, aes(LD1, LD2)) + geom_point(aes(color = Species), size = 3) + theme(text = element_text(size = 20))
```

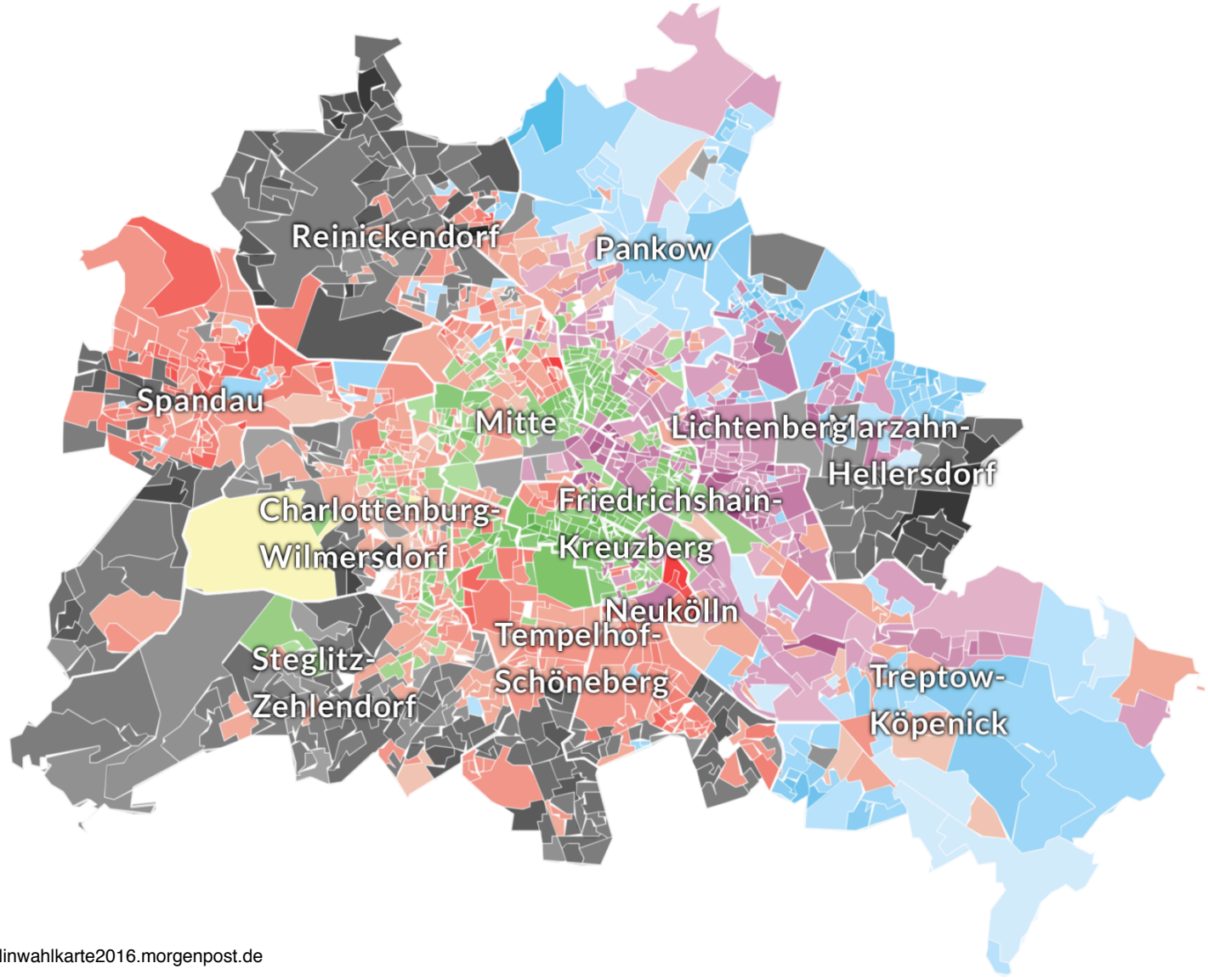


Klassifikationsbäume

Entscheidungsbäume

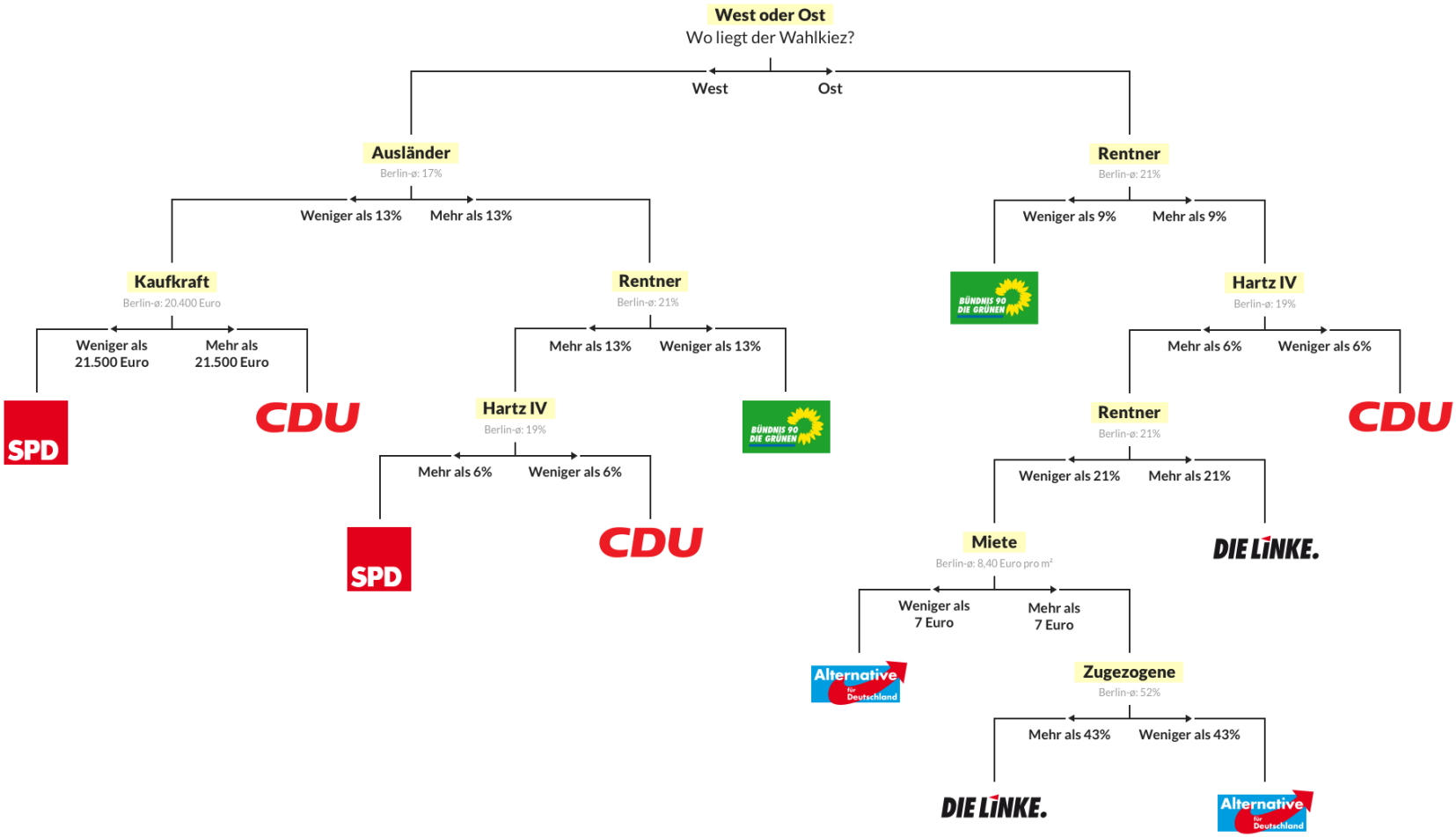
Klassifikationsbäume umfassen das Erstellen einer Reihe von binären Teilungen für die Prädiktorvariablen, um einen Baum zu erstellen, mit dem neue Beobachtungen in eine von zwei oder mehreren Gruppen eingeteilt werden können.

Abgeordnetenhauswahl Berlin 2016



http://berlinwahlkarte2016.morgenpost.de

Abgeordnetenhauswahl Berlin 2016: Entscheidungsbaum



Verfahren

1. Wählen Sie die Prädiktorvariable aus, die die Daten am besten in zwei Gruppen aufteilt, sodass die Reinheit (Homogenität) des Ergebnisses in den beiden Gruppen maximiert wird. Wenn der Prädiktor kontinuierlich ist, wählen Sie einen Schnittpunkt, der die Reinheit für die beiden erstellten Gruppen maximiert. Wenn die Prädiktorvariable kategorisch ist, kombinieren Sie die Kategorien, um zwei Gruppen mit maximaler Reinheit zu erhalten.
2. Trennen Sie die Daten in diese beiden Gruppen und setzen Sie den Vorgang für jede Untergruppe fort.
3. Wiederholen Sie die Schritte 1 und 2, bis eine Untergruppe weniger als eine Mindestanzahl von Beobachtungen enthält oder keine Teilungen die Verunreinigung über einen *festgelegten Schwellenwert* hinaus verringern. Die Untergruppen im endgültigen Satz werden als Endknoten bezeichnet. Jeder Endknoten wird basierend auf dem häufigsten Wert des Ergebnisses für die Stichprobe in diesem Knoten als die eine oder andere Kategorie des Ergebnisses klassifiziert.
4. Um einen Fall zu klassifizieren, führen Sie ihn in der Baumstruktur zu einem Endknoten aus und weisen Sie ihm den in Schritt 3 zugewiesenen modalen Ergebniswert zu.

Classification **A**nd **R**egression **T**ree

Verlustfunktion

$$J(k, t_k) = \frac{m_{\text{left}}}{m} G_{\text{left}} + \frac{m_{\text{right}}}{m} G_{\text{right}}$$

t_k Schwellenwert für das Merkmal k

$m_{\text{left/right}}$ Anzahl Datenpunkte

$G_{\text{left/right}}$ Gini Koeffizient oder Unreinheit (impurity)

$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2$ für den Knoten i und das Merkmal k

Nachteile von Entscheidungsbäumen

algorithmische Komplexität $O(n \times m \log(m))$

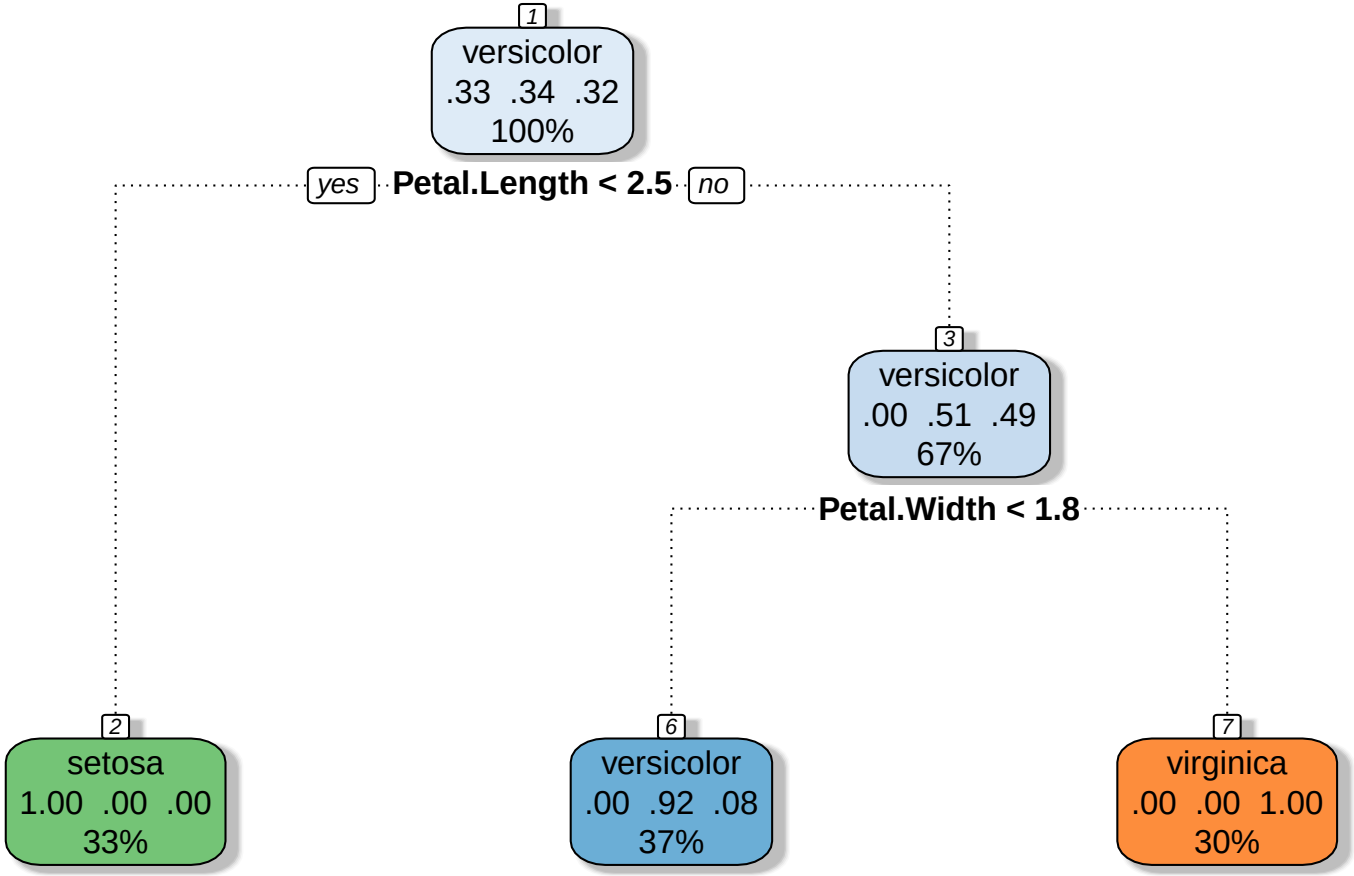
m Anzahl Datenpunkte

n Anzahl Merkmale

- Overfitting (kann z.B. durch Tiefenbegrenzung reduziert werden)
- Sensibel zu kleinen Veränderungen, z.B. Drehungen
- *Random Forest* besteht aus Bäumen, die auf zufällig ausgesuchten Teilmenge der Daten trainiert werden. Am Ende wird die unter allen Bäumen häufigste Vorhersage ausgewählt.

```
library(rpart)
library(gmodels)
library(rpart.plot)
library(rattle)
train <- sample(nrow(iris), 0.7 * nrow(iris))
df.train <- iris[train, ]
df.test <- iris[-train, ]
dtree <- rpart(Species ~ ., data = df.train, method = "class", parms = list(split = "gini"))
```

```
fancyRpartPlot(dtree)
```



Rattle 2021−Jan−12 08:02:01 vitaly

```
(df.test$Species == predict(dtree, df.test, type = "class"))
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
## [25]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [37]  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE  TRUE
```

```
CrossTable(df.test$Species, predict(dtree, df.test, type = "class"))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  45
##
##
##                 | predict(dtree, df.test, type = "class")
## df.test$Species |     setosa | versicolor |  virginica |  Row Total |
## ----------------|------------|------------|------------|------------|
##          setosa |         15 |          0 |          0 |         15 |
##                 |     20.000 |      5.000 |      5.000 |            |
##                 |      1.000 |      0.000 |      0.000 |      0.333 |
##                 |      1.000 |      0.000 |      0.000 |            |
##                 |      0.333 |      0.000 |      0.000 |            |
## ----------------|------------|------------|------------|------------|
##      versicolor |          0 |         13 |          1 |         14 |
##                 |      4.667 |     14.881 |      2.881 |            |
##                 |      0.000 |      0.929 |      0.071 |      0.311 |
##                 |      0.000 |      0.867 |      0.067 |            |
##                 |      0.000 |      0.289 |      0.022 |            |
## ----------------|------------|------------|------------|------------|
##       virginica |          0 |          2 |         14 |         16 |
##                 |      5.333 |      2.083 |     14.083 |            |
##                 |      0.000 |      0.125 |      0.875 |      0.356 |
##                 |      0.000 |      0.133 |      0.933 |            |
##                 |      0.000 |      0.044 |      0.311 |            |
## ----------------|------------|------------|------------|------------|
##    Column Total |         15 |         15 |         15 |         45 |
##                 |      0.333 |      0.333 |      0.333 |            |
## ----------------|------------|------------|------------|------------|
##
##
```

```
CrossTable(df.test$Species == predict(dtree, df.test, type = "class"))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  45
##
##
##            |     FALSE |      TRUE |
##            |-----------|-----------|
##            |         3 |        42 |
##            |     0.067 |     0.933 |
##            |-----------|-----------|
##
##
##
##
```

Confusion matrix

```
library(caret)
confusionMatrix(predict(dtree, df.test, type = "class"), df.test$Species)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         13         2
##   virginica       0          1        14
##
## Overall Statistics
##
##                Accuracy : 0.9333
##                  95% CI : (0.8173, 0.986)
##     No Information Rate : 0.3556
##     P-Value [Acc > NIR] : 5.426e-16
##
##                   Kappa : 0.9
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.9286           0.8750
## Specificity                 1.0000            0.9355           0.9655
## Pos Pred Value              1.0000            0.8667           0.9333
## Neg Pred Value              1.0000            0.9667           0.9333
## Prevalence                  0.3333            0.3111           0.3556
## Detection Rate              0.3333            0.2889           0.3111
## Detection Prevalence        0.3333            0.3333           0.3333
## Balanced Accuracy           1.0000            0.9320           0.9203
```

Qualitätsmaße für Klassifikationsverfahren

- Accuracy

$$\text{Acc} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Spezifität

$$\text{Sp} = \frac{TN}{TN + FP}$$

- Sensitivität (recall)

$$\text{Sn} = \frac{TP}{TP + FN}$$

Die Entscheidungsbäume sind in der Lage nicht offentsichliche Muster in den Daten erkennen.