Aufgabenblatt 3

UK Erweiterungen des linearen Modells

Cordula Eggerth

Matrikelnummer: 00750881

Kursleiter:

Prof. Dr. Marcus Hudec &

Prof. Dr. Wilfried Grossmann

Sommersemester 2019

Aufgabe 1:

Erweitern Sie die in der Vorlesung vorgestellten Analysen zum SPAM-Dataset durch die Verwendung möglichst aller Prädiktoren. Wenden Sie sowohl die Methoden Classification Tree als auch Logistische Regression an. Zeigen Sie die Trennschärfe Ihres Modells mit der ROC-Kurve.

Quelle für die Daten (und deren Beschreibung) zu Aufgabe 1:
https://rdrr.io/cran/kernlab/man/spam.html.

Der Datensatz umfasst 4601 E-Mails, die jeweils entweder Spam oder Non-Spam sind. Insgesamt enthält der Datensatz 2788 E-Mails aus der Kategorie Spam und 1813 E-Mails aus der Kategorie Non-Spam. Zusätzlich zu der Einteilung in Spam und Nicht-Spam werden noch weitere 57 Variablen betrachtet, die vor allem angeben, wie oft bestimmte Wörter und Zeichen im betrachteten E-Mail vorkommen.

Überblick über den Datensatz:

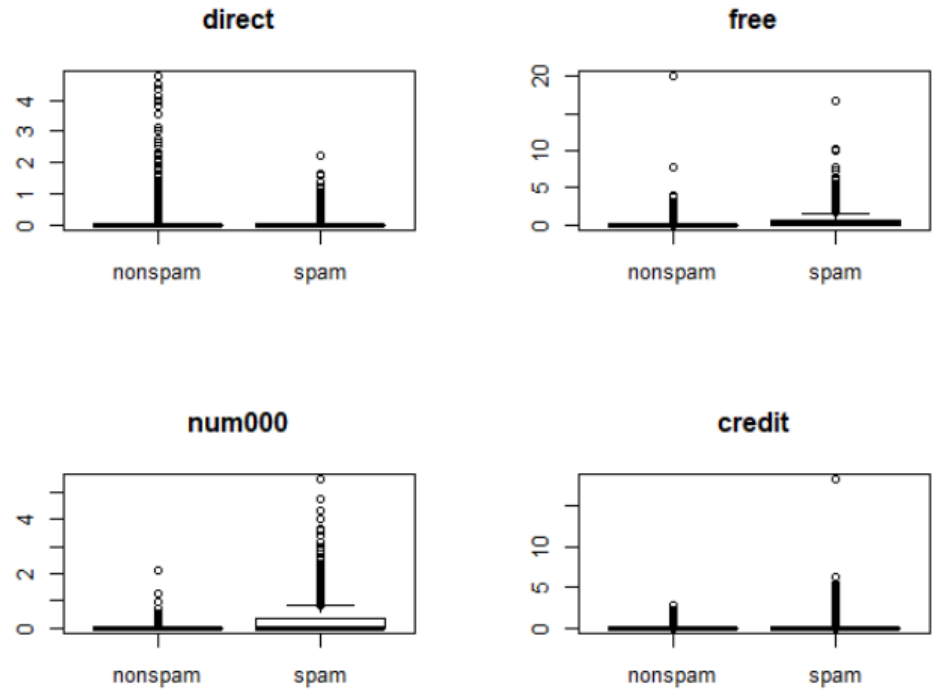
```
> head(spam, n=5)
  make address  all num3d  our over remove internet order mail receive will people report addresses free business
1 0.00    0.64 0.64     0 0.32 0.00   0.00     0.00  0.00 0.00    0.00 0.64   0.00   0.00      0.00 0.32     0.00
2 0.21    0.28 0.50     0 0.14 0.28   0.21     0.07  0.00 0.94    0.21 0.79   0.65   0.21      0.14 0.14     0.07
3 0.06    0.00 0.71     0 1.23 0.19   0.19     0.12  0.64 0.25    0.38 0.45   0.12   0.00      1.75 0.06     0.06
4 0.00    0.00 0.00     0 0.63 0.00   0.31     0.63  0.31 0.63    0.31 0.31   0.31   0.00      0.00 0.31     0.00
5 0.00    0.00 0.00     0 0.63 0.00   0.31     0.63  0.31 0.63    0.31 0.31   0.31   0.00      0.00 0.31     0.00
  email  you credit your font num000 money hp hpl george num650 lab labs telnet num857 data num415 num85
1  1.29 1.93   0.00 0.96    0   0.00  0.00  0   0      0       0   0    0      0      0    0      0     0
2  0.28 3.47   0.00 1.59    0   0.43  0.43  0   0      0       0   0    0      0      0    0      0     0
3  1.03 1.36   0.32 0.51    0   1.16  0.06  0   0      0       0   0    0      0      0    0      0     0
4  0.00 3.18   0.00 0.31    0   0.00  0.00  0   0      0       0   0    0      0      0    0      0     0
5  0.00 3.18   0.00 0.31    0   0.00  0.00  0   0      0       0   0    0      0      0    0      0     0
  technology num1999 parts pm direct cs meeting original project   re  edu table conference charSemicolon
1          0    0.00     0  0   0.00  0       0     0.00       0 0.00 0.00     0          0          0.00
2          0    0.07     0  0   0.00  0       0     0.00       0 0.00 0.00     0          0          0.00
3          0    0.00     0  0   0.06  0       0     0.12       0 0.06 0.06     0          0          0.01
4          0    0.00     0  0   0.00  0       0     0.00       0 0.00 0.00     0          0          0.00
5          0    0.00     0  0   0.00  0       0     0.00       0 0.00 0.00     0          0          0.00
  charRoundbracket charSquarebracket charExclamation charDollar charHash capitalAve capitalLong capitalTotal type
1            0.000                0           0.778      0.000    0.000      3.756          61          278 spam
2            0.132                0           0.372      0.180    0.048      5.114         101         1028 spam
3            0.143                0           0.276      0.184    0.010      9.821         485         2259 spam
4            0.137                0           0.137      0.000    0.000      3.537          40          191 spam
5            0.135                0           0.135      0.000    0.000      3.537          40          191 spam
```

Deskriptive Statistiken zum Datensatz:

```
> summary(spam[spam$type=="nonspam",])
      make              address              all              num3d              our              over        
 Min.   :0.00000   Min.   : 0.0000   Min.   :0.0000   Min.   :0.0000000   Min.   : 0.000   Min.   :0.00000  
 1st Qu.:0.00000   1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.:0.0000000   1st Qu.: 0.000   1st Qu.:0.00000  
 Median :0.00000   Median : 0.0000   Median :0.0000   Median :0.0000000   Median : 0.000   Median :0.00000  
 Mean   :0.07348   Mean   : 0.2445   Mean   :0.2006   Mean   :0.0008859   Mean   : 0.181   Mean   :0.04454  
 3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd Qu.:0.1200   3rd Qu.:0.0000000   3rd Qu.: 0.000   3rd Qu.:0.00000  
 Max.   :4.34000   Max.   :14.2800   Max.   :5.1000   Max.   :0.8700000   Max.   :10.000   Max.   :5.88000  
     remove            internet          order              mail            receive             will        
 Min.   :0.000000   Min.   :0.00000   Min.   :0.00000   Min.   : 0.0000   Min.   :0.00000   Min.   :0.0000  
 1st Qu.:0.000000   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.: 0.0000   1st Qu.:0.00000   1st Qu.:0.0000  
 Median :0.000000   Median :0.00000   Median :0.00000   Median : 0.0000   Median :0.00000   Median :0.0000  
 Mean   :0.009383   Mean   :0.03841   Mean   :0.03805   Mean   : 0.1672   Mean   :0.02171   Mean   :0.5363  
 3rd Qu.:0.000000   3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd Qu.:0.00000   3rd Qu.:0.7525  
 Max.   :3.070000   Max.   :5.88000   Max.   :5.26000   Max.   :18.1800   Max.   :2.00000   Max.   :9.6700  
     people            report          addresses            free              business          email        
 Min.   :0.00000   Min.   : 0.0000   Min.   :0.000000   Min.   : 0.00000   Min.   :0.00000   Min.   :0.00000  
 1st Qu.:0.00000   1st Qu.: 0.0000   1st Qu.:0.000000   1st Qu.: 0.00000   1st Qu.:0.00000   1st Qu.:0.00000  
 Median :0.00000   Median : 0.0000   Median :0.000000   Median : 0.00000   Median :0.00000   Median :0.00000  
 Mean   :0.06166   Mean   : 0.0424   Mean   :0.008318   Mean   : 0.07359   Mean   :0.04835   Mean   :0.09729  
 3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd Qu.:0.000000   3rd Qu.: 0.00000   3rd Qu.:0.00000   3rd Qu.:0.00000  
 Max.   :5.55000   Max.   :10.0000   Max.   :2.240000   Max.   :20.00000   Max.   :3.57000   Max.   :7.69000  
      you              credit             your              font              num000             money        
 Min.   : 0.000   Min.   :0.000000   Min.   : 0.0000   Min.   : 0.00000   Min.   :0.000000   Min.   :0.00000  
 1st Qu.: 0.000   1st Qu.:0.000000   1st Qu.: 0.0000   1st Qu.: 0.00000   1st Qu.:0.000000   1st Qu.:0.00000  
 Median : 0.510   Median :0.000000   Median : 0.0000   Median : 0.00000   Median :0.000000   Median :0.00000  
 Mean   : 1.270   Mean   :0.007579   Mean   : 0.4387   Mean   : 0.04523   Mean   :0.007088   Mean   :0.01714  
 3rd Qu.: 1.992   3rd Qu.:0.000000   3rd Qu.: 0.4600   3rd Qu.: 0.00000   3rd Qu.:0.000000   3rd Qu.:0.00000  
 Max.   :18.750   Max.   :2.700000   Max.   :10.7100   Max.   :11.42000   Max.   :2.120000   Max.   :9.75000  
```

```
       hp                hpl              george            num650       
 Min.   : 0.0000   Min.   : 0.000   Min.   : 0.0000   Min.   :0.0000  
 1st Qu.: 0.0000   1st Qu.: 0.000   1st Qu.: 0.0000   1st Qu.:0.0000  
 Median : 0.0000   Median : 0.000   Median : 0.0000   Median :0.0000  
 Mean   : 0.8955   Mean   : 0.432   Mean   : 1.2653   Mean   :0.1938  
 3rd Qu.: 1.0000   3rd Qu.: 0.330   3rd Qu.: 0.1625   3rd Qu.:0.0000  
 Max.   :20.8300   Max.   :16.660   Max.   :33.3300   Max.   :5.8800  
      lab               labs             telnet            num857       
 Min.   : 0.0000   Min.   :0.0000   Min.   : 0.000   Min.   :0.00000  
 1st Qu.: 0.0000   1st Qu.:0.0000   1st Qu.: 0.000   1st Qu.:0.00000  
 Median : 0.0000   Median :0.0000   Median : 0.000   Median :0.00000  
 Mean   : 0.1628   Mean   :0.1659   Mean   : 0.106   Mean   :0.07731  
 3rd Qu.: 0.0000   3rd Qu.:0.0000   3rd Qu.: 0.000   3rd Qu.:0.00000  
 Max.   :14.2800   Max.   :5.8800   Max.   :12.500   Max.   :4.76000  
      data              num415            num85          technology     
 Min.   : 0.000   Min.   :0.00000   Min.   : 0.0000   Min.   :0.0000  
 1st Qu.: 0.000   1st Qu.:0.00000   1st Qu.: 0.0000   1st Qu.:0.0000  
 Median : 0.000   Median :0.00000   Median : 0.0000   Median :0.0000  
 Mean   : 0.151   Mean   :0.07779   Mean   : 0.1695   Mean   :0.1417  
 3rd Qu.: 0.000   3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd Qu.:0.0000  
 Max.   :18.180   Max.   :4.76000   Max.   :20.0000   Max.   :7.6900  
    num1999           parts              pm              direct       
 Min.   :0.0000   Min.   :0.00000   Min.   : 0.0000   Min.   :0.00000  
 1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.: 0.0000   1st Qu.:0.00000  
 Median :0.0000   Median :0.00000   Median : 0.0000   Median :0.00000  
 Mean   :0.1977   Mean   :0.01872   Mean   : 0.1217   Mean   :0.08312  
 3rd Qu.:0.1000   3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd Qu.:0.00000  
 Max.   :6.8900   Max.   :8.33000   Max.   :11.1100   Max.   :4.76000  
       cs               meeting          original          project       
 Min.   :0.00000   Min.   : 0.0000   Min.   :0.00000   Min.   : 0.0000  
 1st Qu.:0.00000   1st Qu.: 0.0000   1st Qu.:0.00000   1st Qu.: 0.0000  
 Median :0.00000   Median : 0.0000   Median :0.00000   Median : 0.0000  
 Mean   :0.07203   Mean   : 0.2168   Mean   :0.07058   Mean   : 0.1266  
 3rd Qu.:0.00000   3rd Qu.: 0.0000   3rd Qu.:0.00000   3rd Qu.: 0.0000  
 Max.   :7.14000   Max.   :14.2800   Max.   :3.57000   Max.   :20.0000  
       re                edu              table            conference      
 Min.   : 0.0000   Min.   : 0.0000   Min.   :0.000000   Min.   : 0.00000  
 1st Qu.: 0.0000   1st Qu.: 0.0000   1st Qu.:0.000000   1st Qu.: 0.00000  
 Median : 0.0000   Median : 0.0000   Median :0.000000   Median : 0.00000  
 Mean   : 0.4158   Mean   : 0.2872   Mean   :0.008192   Mean   : 0.05123  
 3rd Qu.: 0.3125   3rd Qu.: 0.0000   3rd Qu.:0.000000   3rd Qu.: 0.00000  
 Max.   :21.4200   Max.   :22.0500   Max.   :2.170000   Max.   :10.00000  
 charSemicolon     charRoundbracket  charSquarebracket charExclamation  
 Min.   :0.00000   Min.   :0.0000    Min.   :0.00000   Min.   : 0.000  
 1st Qu.:0.00000   1st Qu.:0.0000    1st Qu.:0.00000   1st Qu.: 0.000  
 Median :0.00000   Median :0.0645    Median :0.00000   Median : 0.000  
 Mean   :0.05028   Mean   :0.1586    Mean   :0.02268   Mean   : 0.110  
 3rd Qu.:0.00000   3rd Qu.:0.2220    3rd Qu.:0.00000   3rd Qu.: 0.027  
 Max.   :4.38500   Max.   :5.2770    Max.   :4.08100   Max.   :32.478  
  charDollar        charHash       capitalAve       capitalLong      
 Min.   :0.00000   Min.   :0.00000   Min.   :  1.000   Min.   :   1.00  
 1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:  1.384   1st Qu.:   4.00  
 Median :0.00000   Median :0.00000   Median :  1.857   Median :  10.00  
 Mean   :0.01165   Mean   :0.02171   Mean   :  2.377   Mean   :  18.21  
 3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:  2.555   3rd Qu.:  18.00  
 Max.   :2.03800   Max.   :7.40700   Max.   :251.000   Max.   :1488.00  
  capitalTotal        type        
 Min.   :   1.00   nonspam:2788  
 1st Qu.:  18.75   spam   :   0  
 Median :  54.00                 
 Mean   : 161.47                 
 3rd Qu.: 141.00                 
 Max.   :5902.00                 
```

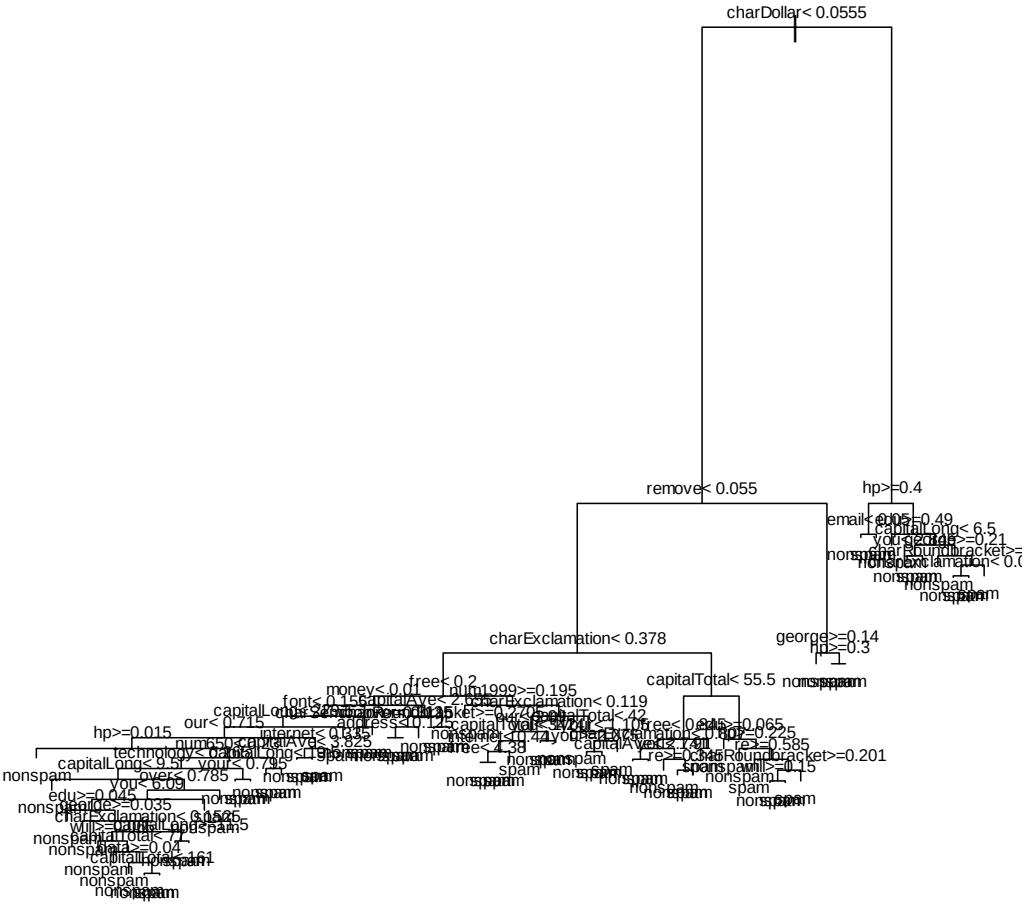
Boxplots für die Variablen `direct`, `free`, `num000`, `credit` aufgeteilt nach Kategorie `spam` und `nonspam`:



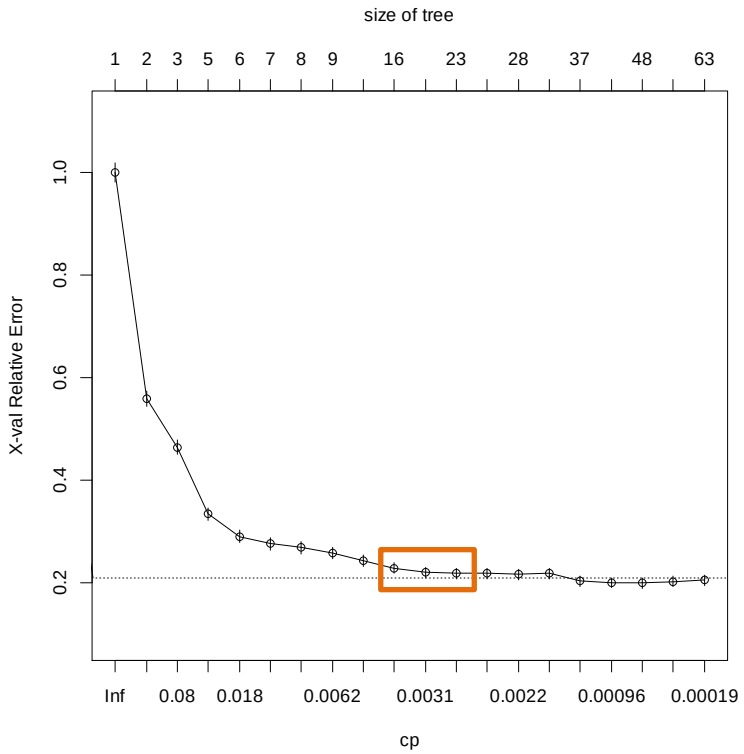
Zur Untersuchung des Datensatzes wurden mittels **Recursive Partioning** verschiedene **Classification Trees** (unter Einbeziehung sämtlicher Variablen) erstellt.

Fall 1 (cp=0.0001):

Da der cp-Wert, also der Cost-Complexity-Parameter, sehr klein ist, ist der Classification Tree sehr groß und umfasst viele Variablen und Splits. Wie die folgenden Plots zeigen, erreicht der `xerror`, also der durch Cross-Validation erzielte Fehler des Trees, bereits bei einem cp von ca. 0.003 einen annehmbaren Wert und fällt dann nicht mehr so stark ab (siehe orange Markierung in den untenstehenden R-Outputs/Plots).

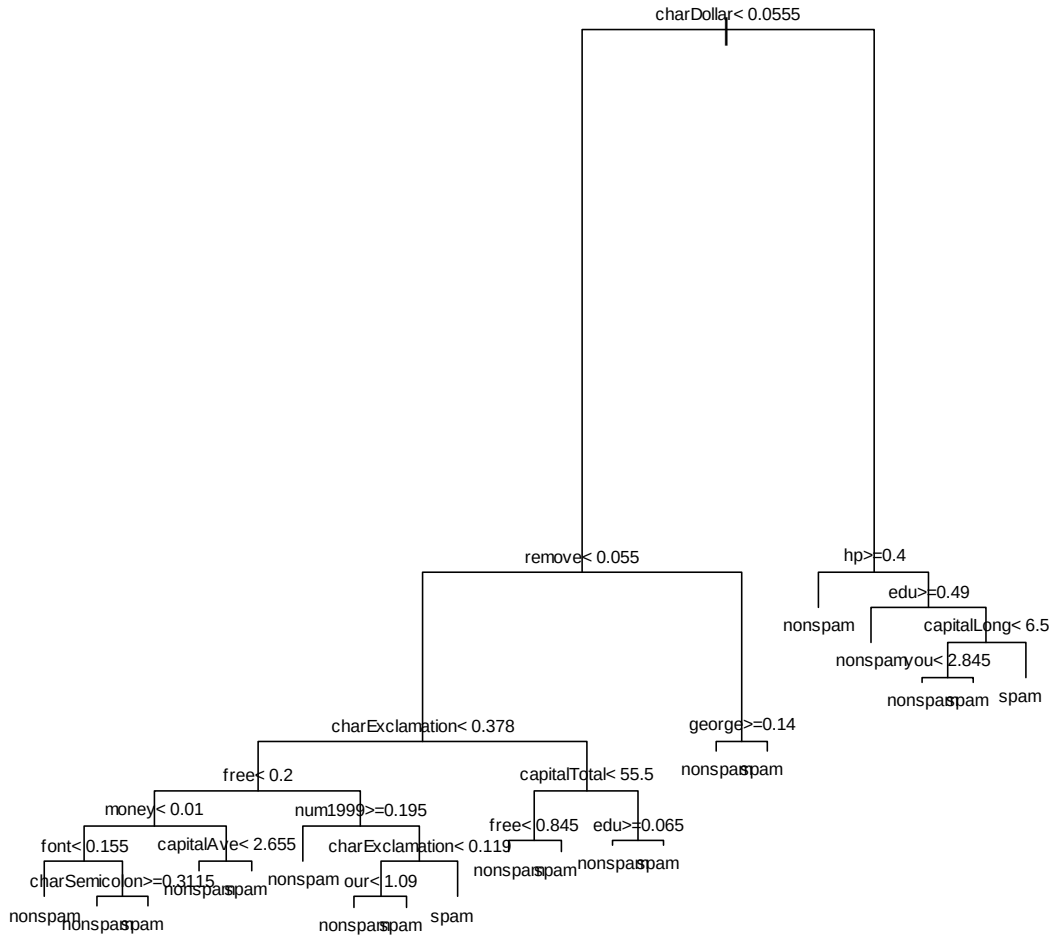


```
> printcp(spam.rp1)

Classification tree:
rpart(formula = factor.type ~ make + address + all + num3d +
    our + over + remove + internet + order + mail + receive +
    will + people + report + addresses + free + business + email +
    you + credit + your + font + num000 + money + hp + hpl +
    george + num650 + lab + labs + telnet + num857 + data + num415 +
    num85 + technology + num1999 + parts + pm + direct + cs +
    meeting + original + project + re + edu + table + conference +
    charSemicolon + charRoundbracket + charSquarebracket + charExclamation +
    charDollar + charHash + capitalAve + capitalLong + capitalTotal,
    cp = 1e-04)

Variables actually used in tree construction:
 [1] address           capitalAve        capitalLong       capitalTotal      charDollar        charExclamation
 [7] charRoundbracket charSemicolon     data              edu               email             font
[13] free              george            hp                internet          money             num1999
[19] num650            our               over              re                remove            technology
[25] will              you               your

Root node error: 1813/4601 = 0.39404

n= 4601

           CP nsplit rel error  xerror     xstd
1  0.47655819      0   1.00000 1.00000 0.018282
2  0.14892443      1   0.52344 0.55929 0.015508
3  0.04302261      2   0.37452 0.46442 0.014467
4  0.03088803      4   0.28847 0.33425 0.012652
5  0.01047987      5   0.25758 0.29068 0.011915
6  0.00827358      6   0.24710 0.27634 0.011654
7  0.00717044      7   0.23883 0.26862 0.011510
8  0.00529509      8   0.23166 0.25758 0.011298
9  0.00441258     14   0.19581 0.24269 0.011003
10 0.00358522     15   0.19140 0.22890 0.010718
11 0.00275786     19   0.17705 0.22008 0.010529
12 0.00257400     22   0.16878 0.21842 0.010493
13 0.00220629     25   0.16106 0.21842 0.010493
14 0.00211436     27   0.15665 0.21622 0.010445
15 0.00165472     33   0.14396 0.21842 0.010493
16 0.00110314     36   0.13900 0.20353 0.010162
17 0.00082736     43   0.13127 0.20022 0.010086
18 0.00055157     47   0.12796 0.19967 0.010073
19 0.00036771     53   0.12466 0.20243 0.010136
20 0.00010000     62   0.12135 0.20518 0.010199
```



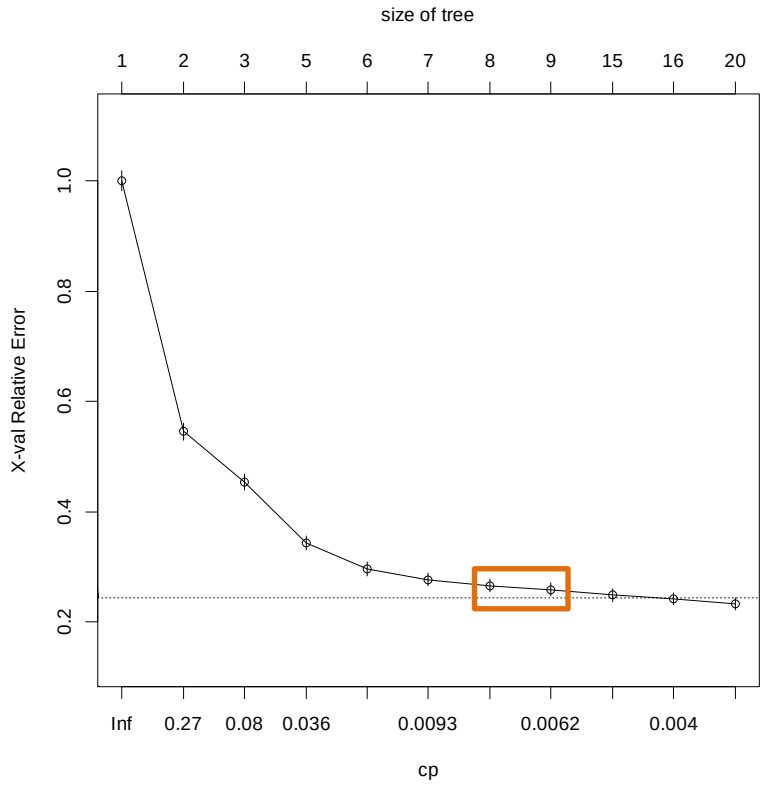
Fall 2 (cp=0.003):

Da der cp-Wert, also der Cost-Complexity-Parameter, nun größer ist, ist der Classification Tree kleiner und umfasst nicht mehr so viele Variablen und Splits wie im Fall 1. Wie die folgenden Plots zeigen, erreicht der `xerror` bei einem cp von ca. 0.009 einen annehmbaren Wert.



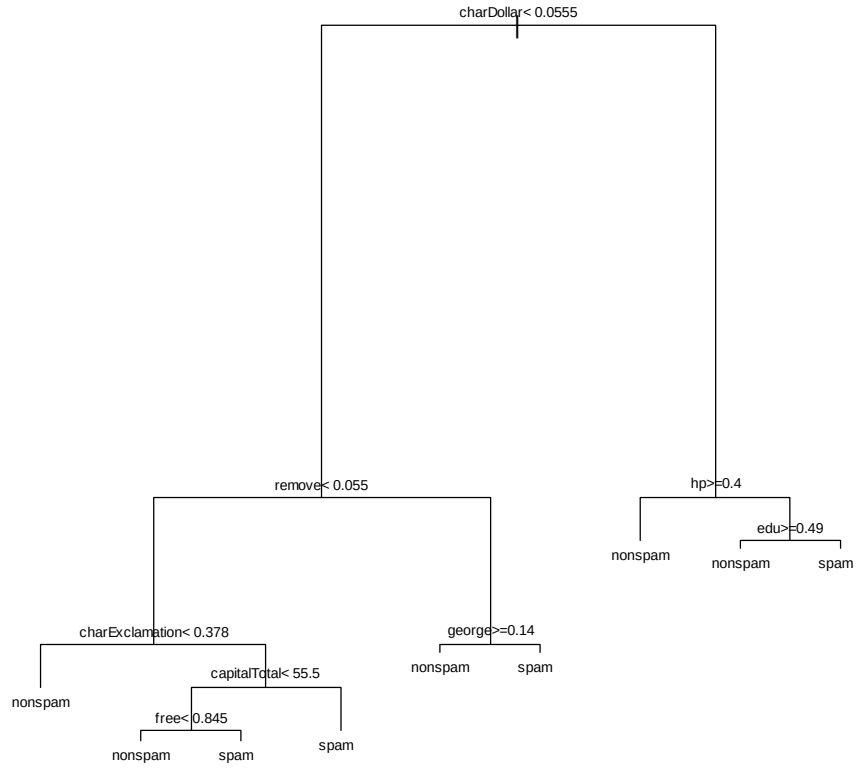

```
> printcp(spam.rp2)

Classification tree:
rpart(formula = factor.type ~ make + address + all + num3d +
    our + over + remove + internet + order + mail + receive +
    will + people + report + addresses + free + business + email +
    you + credit + your + font + num000 + money + hp + hpl +
    george + num650 + lab + labs + telnet + num857 + data + num415 +
    num85 + technology + num1999 + parts + pm + direct + cs +
    meeting + original + project + re + edu + table + conference +
    charSemicolon + charRoundbracket + charSquarebracket + charExclamation +
    charDollar + charHash + capitalAve + capitalLong + capitalTotal,
    cp = 0.003)

Variables actually used in tree construction:
 [1] capitalAve      capitalLong     capitalTotal    charDollar      charExclamation charSemicolon
 [7] edu             font            free            george          hp              money
[13] num1999         our             remove          you

Root node error: 1813/4601 = 0.39404

n= 4601
```

	CP	nsplit	rel error	xerror	xstd
1	0.4765582	0	1.00000	1.00000	0.018282
2	0.1489244	1	0.52344	0.54550	0.015369
3	0.0430226	2	0.37452	0.45339	0.014332
4	0.0308880	4	0.28847	0.34418	0.012810
5	0.0104799	5	0.25758	0.29675	0.012022
6	0.0082736	6	0.24710	0.27689	0.011665
7	0.0071704	7	0.23883	0.26641	0.011468
8	0.0052951	8	0.23166	0.25924	0.011331
9	0.0044126	14	0.19581	0.24876	0.011125
10	0.0035852	15	0.19140	0.24159	0.010980
11	0.0030000	19	0.17705	0.23331	0.010810

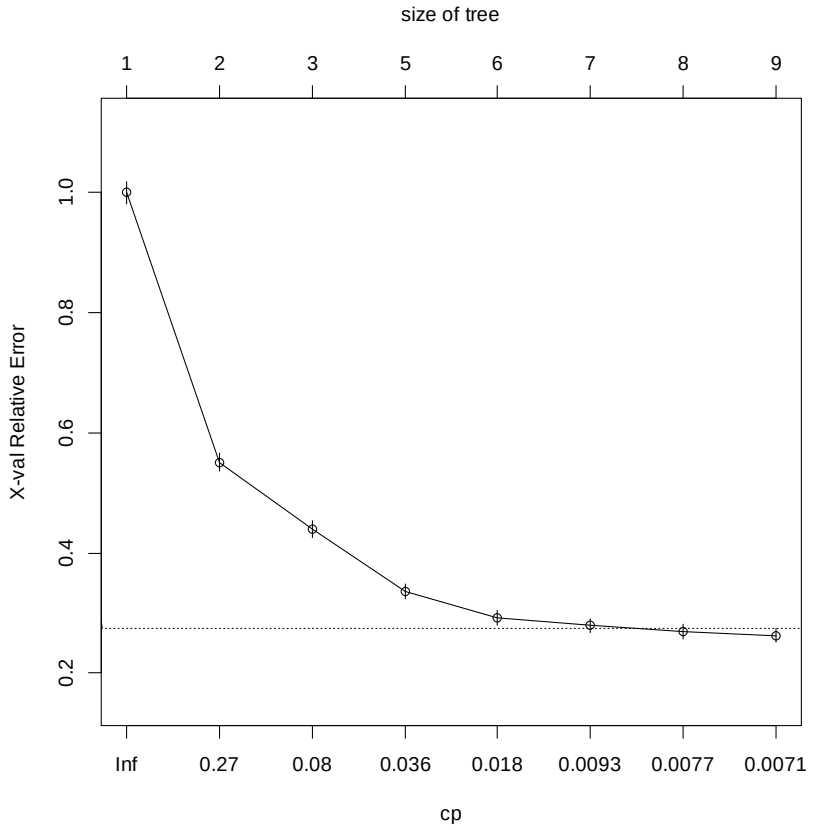


Fall 3 (cp=0.007):

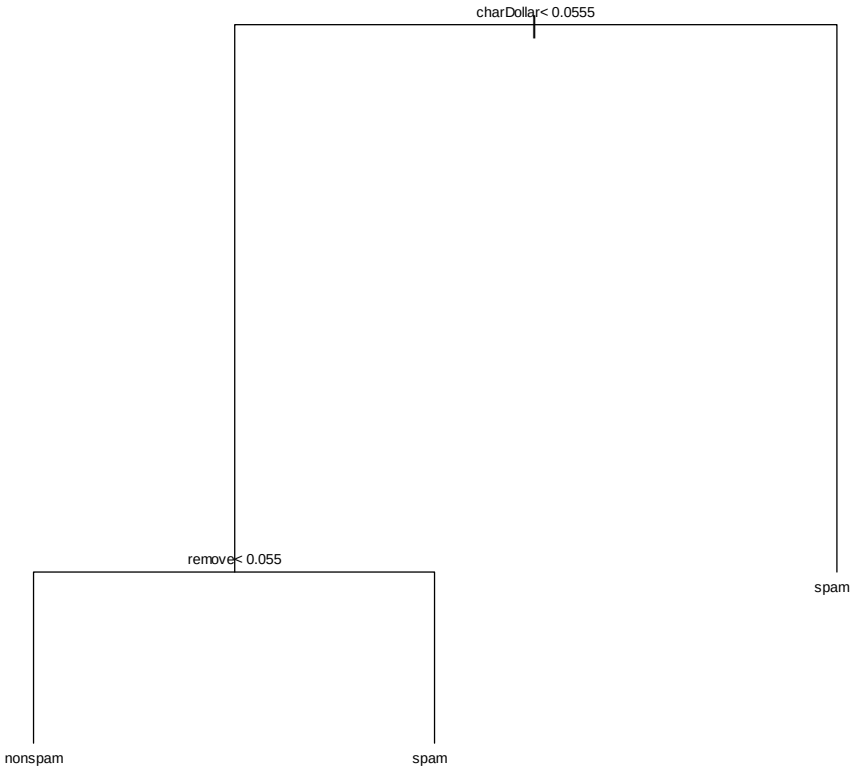
Da der cp-Wert, nun noch größer ist, ist der Classification Tree noch kleiner.



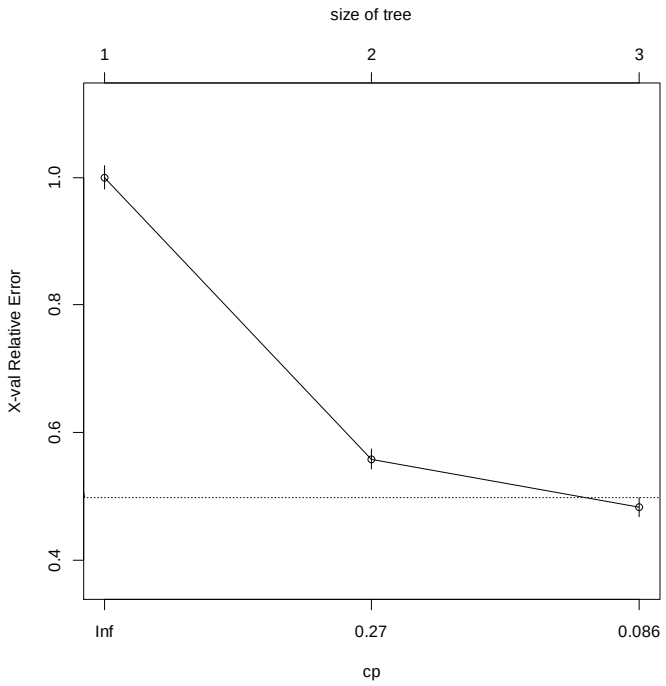

```
> printcp(spam.rp3)

Classification tree:
rpart(formula = factor.type ~ make + address + all + num3d +
    our + over + remove + internet + order + mail + receive +
    will + people + report + addresses + free + business + email +
    you + credit + your + font + num000 + money + hp + hpl +
    george + num650 + lab + labs + telnet + num857 + data + num415 +
    num85 + technology + num1999 + parts + pm + direct + cs +
    meeting + original + project + re + edu + table + conference +
    charSemicolon + charRoundbracket + charSquarebracket + charExclamation +
    charDollar + charHash + capitalAve + capitalLong + capitalTotal,
    cp = 0.007)

Variables actually used in tree construction:
[1] capitalTotal    charDollar      charExclamation edu             free            george
[7] hp              remove

Root node error: 1813/4601 = 0.39404

n= 4601

         CP nsplit rel error  xerror     xstd
1 0.4765582      0   1.00000 1.00000 0.018282
2 0.1489244      1   0.52344 0.55102 0.015425
3 0.0430226      2   0.37452 0.43960 0.014159
4 0.0308880      4   0.28847 0.33646 0.012688
5 0.0104799      5   0.25758 0.29233 0.011944
6 0.0082736      6   0.24710 0.27910 0.011705
7 0.0071704      7   0.23883 0.26917 0.011520
8 0.0070000      8   0.23166 0.26255 0.011394
```



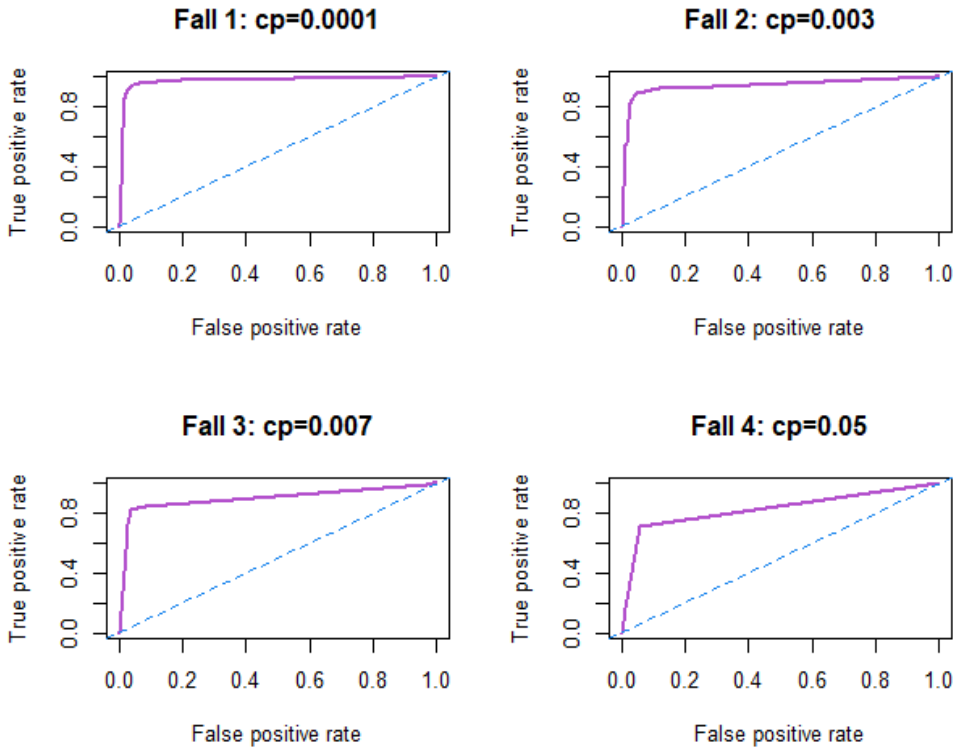
Fall 4 (cp=0.05):
Da der cp-Wert, nun noch größer ist, ist der Classification Tree noch kleiner.



```
> printcp(spam.rp4)

Classification tree:
rpart(formula = factor.type ~ make + address + all + num3d +
    our + over + remove + internet + order + mail + receive +
    will + people + report + addresses + free + business + email +
    you + credit + your + font + num000 + money + hp + hpl +
    george + num650 + lab + labs + telnet + num857 + data + num415 +
    num85 + technology + num1999 + parts + pm + direct + cs +
    meeting + original + project + re + edu + table + conference +
    charSemicolon + charRoundbracket + charSquarebracket + charExclamation +
    charDollar + charHash + capitalAve + capitalLong + capitalTotal,
    cp = 0.05)

Variables actually used in tree construction:
[1] charDollar remove

Root node error: 1813/4601 = 0.39404

n= 4601

        CP nsplit rel error  xerror     xstd
1 0.47656      0   1.00000 1.00000 0.018282
2 0.14892      1   0.52344 0.55819 0.015497
3 0.05000      2   0.37452 0.48263 0.014683
```



Um die **Trennschärfe der Classification Trees** (je nach gewähltem cp-Wert) zu zeigen, wurden **ROC-Kurven** angefertigt, wie in den folgenden Plots ersichtlich ist. Im Fall 1 ist der cp-Wert sehr klein, und die violette ROC-Kurve sehr weit im oberen linken Eck gelegen, was bedeutet, dass die Trennschärfe sehr gut ist. Hingegen ist die Trennschärfe im Fall 4 bei einem cp von 0.05 nicht mehr so hoch, d.h. sie liegt weiter unten in Richtung der gestrichelten Linie, die den Fall der zufälligen Einteilung der Klassen darstellt.



Abgesehen von Classification Trees wurde für den Datensatz eine **logistische Regression** sowohl mit Logit- als auch mit Probit-Linkfunktion durchgeführt.

Modelloutput für die logistische Regression (mit `link=logit`):

```
> summary(res.logit)

Call:
glm(formula = factor.type ~ make + address + all + num3d + our +
    over + remove + internet + order + mail + receive + will +
    people + report + addresses + free + business + email + you +
    credit + your + font + num000 + money + hp + hpl + george +
    num650 + lab + labs + telnet + num857 + data + num415 + num85 +
    technology + num1999 + parts + pm + direct + cs + meeting +
    original + project + re + edu + table + conference + charSemicolon +
    charRoundbracket + charSquarebracket + charExclamation +
    charDollar + charHash + capitalAve + capitalLong + capitalTotal,
    family = binomial(link = logit))

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-4.127  -0.203   0.000   0.114   5.364
```

```
Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.569e+00  1.420e-01 -11.044  < 2e-16 ***
make            -3.895e-01  2.315e-01  -1.683 0.092388 .
address         -1.458e-01  6.928e-02  -2.104 0.035362 *
all              1.141e-01  1.103e-01   1.035 0.300759
num3d            2.252e+00  1.507e+00   1.494 0.135168
our              5.624e-01  1.018e-01   5.524 3.31e-08 ***
over             8.830e-01  2.498e-01   3.534 0.000409 ***
remove           2.279e+00  3.328e-01   6.846 7.57e-12 ***
internet         5.696e-01  1.682e-01   3.387 0.000707 ***
order            7.343e-01  2.849e-01   2.577 0.009958 **
mail             1.275e-01  7.262e-02   1.755 0.079230 .
receive         -2.557e-01  2.979e-01  -0.858 0.390655
will            -1.383e-01  7.405e-02  -1.868 0.061773 .
people          -7.961e-02  2.303e-01  -0.346 0.729557
report           1.447e-01  1.364e-01   1.061 0.288855
addresses        1.236e+00  7.254e-01   1.704 0.088370 .
free             1.039e+00  1.457e-01   7.128 1.01e-12 ***
business         9.599e-01  2.251e-01   4.264 2.01e-05 ***
email            1.203e-01  1.172e-01   1.027 0.304533
you              8.131e-02  3.505e-02   2.320 0.020334 *
credit           1.047e+00  5.383e-01   1.946 0.051675 .
your             2.419e-01  5.243e-02   4.615 3.94e-06 ***
font             2.013e-01  1.627e-01   1.238 0.215838
num000           2.245e+00  4.714e-01   4.762 1.91e-06 ***
money            4.264e-01  1.621e-01   2.630 0.008535 **
hp              -1.920e+00  3.128e-01  -6.139 8.31e-10 ***
hpl             -1.040e+00  4.396e-01  -2.366 0.017966 *
george          -1.177e+01  2.113e+00  -5.569 2.57e-08 ***
num650           4.454e-01  1.991e-01   2.237 0.025255 *
lab             -2.486e+00  1.502e+00  -1.656 0.097744 .
labs            -3.299e-01  3.137e-01  -1.052 0.292972
telnet          -1.702e-01  4.815e-01  -0.353 0.723742
num857           2.549e+00  3.283e+00   0.776 0.437566
data            -7.383e-01  3.117e-01  -2.369 0.017842 *
num415           6.679e-01  1.601e+00   0.417 0.676490
num85           -2.055e+00  7.883e-01  -2.607 0.009124 **
technology       9.237e-01  3.091e-01   2.989 0.002803 **
num1999          4.651e-02  1.754e-01   0.265 0.790819
parts           -5.968e-01  4.232e-01  -1.410 0.158473
pm              -8.650e-01  3.828e-01  -2.260 0.023844 *
direct          -3.046e-01  3.636e-01  -0.838 0.402215
cs              -4.505e+01  2.660e+01  -1.694 0.090333 .
meeting         -2.689e+00  8.384e-01  -3.207 0.001342 **
original        -1.247e+00  8.064e-01  -1.547 0.121978
project         -1.573e+00  5.292e-01  -2.973 0.002953 **
re              -7.923e-01  1.556e-01  -5.091 3.56e-07 ***
edu             -1.459e+00  2.686e-01  -5.434 5.52e-08 ***
table           -2.326e+00  1.659e+00  -1.402 0.160958
conference      -4.016e+00  1.611e+00  -2.493 0.012672 *
charSemicolon   -1.291e+00  4.422e-01  -2.920 0.003503 **
charRoundbracket -1.881e-01  2.494e-01  -0.754 0.450663
charSquarebracket -6.574e-01  8.383e-01  -0.784 0.432914
charExclamation  3.472e-01  8.926e-02   3.890 0.000100 ***
charDollar       5.336e+00  7.064e-01   7.553 4.24e-14 ***
charHash         2.403e+00  1.113e+00   2.159 0.030883 *
capitalAve       1.199e-02  1.884e-02   0.636 0.524509
capitalLong      9.118e-03  2.521e-03   3.618 0.000297 ***
capitalTotal     8.437e-04  2.251e-04   3.747 0.000179 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 6170.2  on 4600  degrees of freedom
Residual deviance: 1815.8  on 4543  degrees of freedom
AIC: 1931.8

Number of Fisher Scoring iterations: 13
```

Im Modell sind die auf dem 0.001-Niveau signifikanten Variablen (z.B. `num000`, `free`, `business`) mit 3 Sternen gekennzeichnet, und die auf dem 0.01-Niveau signifikanten Variablen (z.B. `money`, `order`) mit 2 Sternen. Es wäre empfehlenswert, diese Variablen im Modell zu behalten. Bei `link=probit` ist die Vorgehensweise dieselbe, es wird nur eine andere Link-Funktion verwendet.

Modelloutput für die logistische Regression (mit `link=probit`):

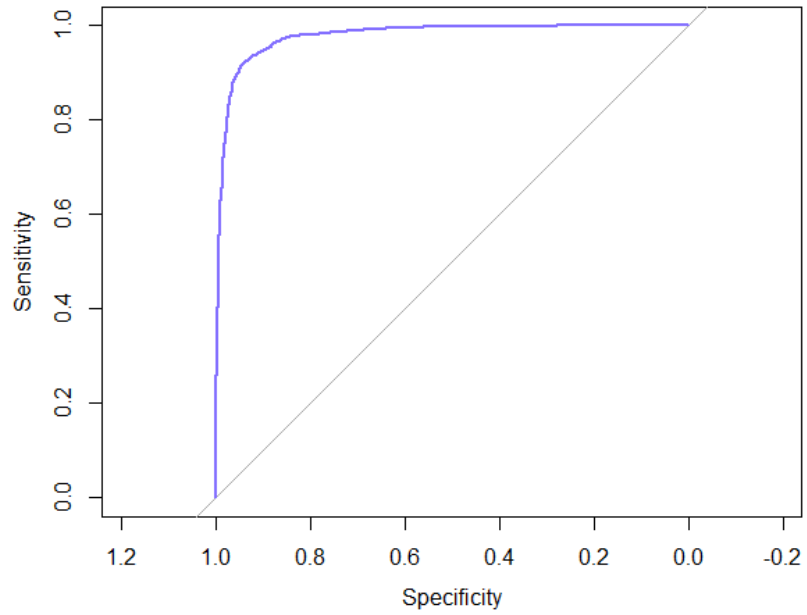
```
> summary(res.probit)

Call:
glm(formula = factor.type ~ make + address + all + num3d + our +
    over + remove + internet + order + mail + receive + will +
    people + report + addresses + free + business + email + you +
    credit + your + font + num000 + money + hp + hpl + george +
    num650 + lab + labs + telnet + num857 + data + num415 + num85 +
    technology + num1999 + parts + pm + direct + cs + meeting +
    original + project + re + edu + table + conference + charSemicolon +
    charRoundbracket + charSquarebracket + charExclamation +
    charDollar + charHash + capitalAve + capitalLong + capitalTotal,
    family = binomial(link = probit))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.0924  -0.2042   0.0000   0.1176   4.7895

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    -8.372e-01  7.460e-02 -11.222  < 2e-16 ***
make           -1.752e-01  1.143e-01  -1.533 0.125273
address        -8.573e-02  3.268e-02  -2.624 0.008703 **
all             1.132e-01  6.122e-02   1.849 0.064510 .
num3d           1.365e+00  7.531e-01   1.813 0.069861 .
our             3.136e-01  5.335e-02   5.878 4.16e-09 ***
over            4.680e-01  1.260e-01   3.713 0.000204 ***
remove          9.861e-01  1.361e-01   7.245 4.34e-13 ***
internet        2.810e-01  8.370e-02   3.358 0.000786 ***
order           3.037e-01  1.413e-01   2.150 0.031573 *
mail            7.886e-02  4.132e-02   1.909 0.056295 .
receive        -7.764e-02  1.561e-01  -0.497 0.618956
will           -8.670e-02  3.984e-02  -2.176 0.029547 *
people         -2.889e-02  1.203e-01  -0.240 0.810282
report          1.093e-01  8.021e-02   1.363 0.172972
addresses       8.332e-01  3.813e-01   2.185 0.028889 *
free            5.131e-01  6.990e-02   7.340 2.14e-13 ***
business        4.645e-01  1.120e-01   4.148 3.36e-05 ***
email           1.038e-01  6.458e-02   1.608 0.107864
you             3.916e-02  1.925e-02   2.035 0.041866 *
```

```
credit             4.395e-01  2.095e-01   2.098 0.035907 *
your               1.496e-01  2.812e-02   5.320 1.04e-07 ***
font               1.440e-01  8.207e-02   1.754 0.079419 .
num000             1.301e+00  2.282e-01   5.700 1.20e-08 ***
money              2.185e-01  7.741e-02   2.822 0.004766 **
hp                -7.824e-01  1.133e-01  -6.908 4.92e-12 ***
hpl               -7.014e-01  2.061e-01  -3.404 0.000665 ***
george            -3.488e+00  7.797e-01  -4.474 7.67e-06 ***
num650             2.453e-01  9.928e-02   2.470 0.013493 *
lab               -1.453e+00  7.628e-01  -1.904 0.056877 .
labs              -2.264e-01  1.735e-01  -1.305 0.191819
telnet            -9.974e-02  2.227e-01  -0.448 0.654190
num857             1.156e+00  1.451e+00   0.797 0.425639
data              -4.165e-01  1.619e-01  -2.573 0.010088 *
num415            -2.776e-01  8.676e-01  -0.320 0.748973
num85             -1.248e+00  4.214e-01  -2.962 0.003053 **
technology         4.618e-01  1.610e-01   2.869 0.004117 **
num1999            1.755e-03  1.013e-01   0.017 0.986168
parts             -2.974e-01  2.099e-01  -1.417 0.156573
pm                -4.683e-01  1.979e-01  -2.366 0.017974 *
direct            -1.444e-01  1.894e-01  -0.762 0.446028
cs                -2.422e+01  1.236e+01  -1.959 0.050072 .
meeting           -1.502e+00  4.228e-01  -3.552 0.000382 ***
original          -6.513e-01  3.910e-01  -1.666 0.095733 .
project           -8.124e-01  2.548e-01  -3.189 0.001430 **
re                -4.053e-01  7.396e-02  -5.480 4.25e-08 ***
edu               -6.723e-01  1.224e-01  -5.494 3.92e-08 ***
table             -1.223e+00  7.201e-01  -1.698 0.089537 .
conference        -2.005e+00  7.142e-01  -2.807 0.005003 **
charSemicolon     -8.082e-01  2.342e-01  -3.450 0.000560 ***
charRoundbracket  -1.316e-01  1.410e-01  -0.934 0.350489
charSquarebracket -4.073e-01  4.313e-01  -0.944 0.344959
charExclamation    1.721e-01  3.110e-02   5.532 3.16e-08 ***
charDollar         2.235e+00  3.307e-01   6.758 1.40e-11 ***
charHash           1.316e+00  3.190e-01   4.124 3.72e-05 ***
capitalAve        -2.712e-03  8.735e-03  -0.310 0.756210
capitalLong        4.215e-03  1.269e-03   3.321 0.000896 ***
capitalTotal       4.959e-04  1.160e-04   4.274 1.92e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1


(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 6170.2  on 4600  degrees of freedom
Residual deviance: 1910.5  on 4543  degrees of freedom
AIC: 2026.5

Number of Fisher Scoring iterations: 25
```

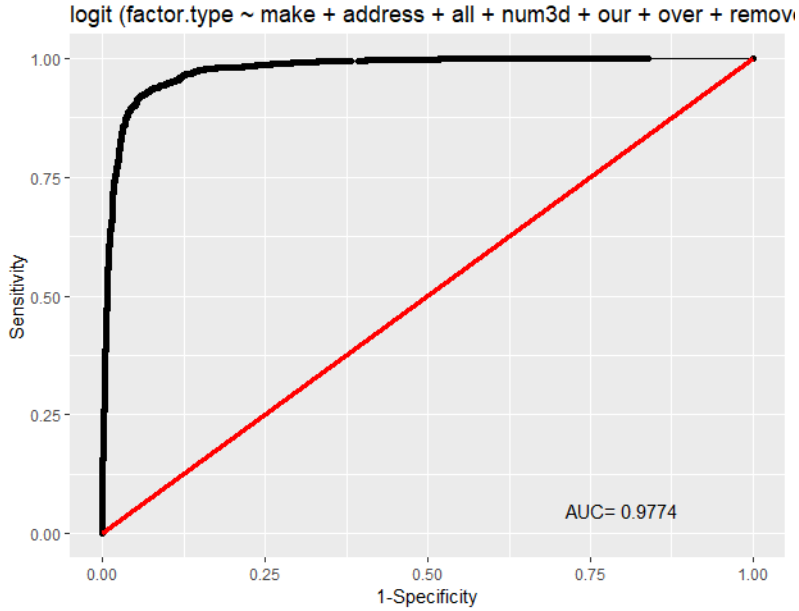
Für die logistische Regression (`link=logit`) wurde die **ROC-Kurve** auf 2 Arten dargestellt, wobei die Fläche unter der Kurve idealerweise möglichst groß sein sollte. In diesem Beispiel beträgt sie ca. 97%, d.h. das Modell bietet also eine gute Trennschärfe.

ROC-Kurven für die logistische Regression mit link=logit:

```
# methode 1:
par(mfrow=c(1,1))
prob.pred = predict(res.logit,type=c("response"))
spam$prob.pred=prob.pred
roc.1 <- roc(factor.type ~ prob.pred, data=spam)
plot(roc.1, col="lightslateblue")
```




```
# methode 2:
modelfit <- glm(formula=factor.type ~ make+address+all+num3d+our+over+remove+
                  internet+order+mail+receive+will+people+
                  report+addresses+free+business+email+you+
                  credit+your+font+num000+money+hp+hpl+george+
                  num650+lab+labs+telnet+num857+data+num415+
                  num85+technology+num1999+parts+pm+direct+cs+
                  meeting+original+project+re+edu+table+conference+
                  charSemicolon+charRoundbracket+charSquarebracket+
                  charExclamation+charDollar+charHash+capitalAve+
                  capitalLong+capitalTotal,
                family=binomial(), data=spam, na.action=na.omit)
rocplot(modelfit)
```



R-Code zu Aufgabe 1:

```
rm(list=ls())

install.packages("kernlab")
install.packages("rpart")
install.packages("verification")
install.packages("ROCR")
install.packages("pROC")
install.packages("Deducer")

library(kernlab)
library(ggplot2)
library("MASS")
library("rpart")
library("verification")
library(class)
library(ROCR)
library(pROC)
library(Deducer)

setwd("C:/Users/Coala/Desktop/A3_ERWEIT")

#*********************************************************************************************
# AUFGABE 1
#*********************************************************************************************
# Erweitern Sie die in der Vorlesung vorgestellten Analysen zum SPAM-Dataset durch die
# Verwendung möglichst aller Prädiktoren.
# Wenden Sie sowohl die Methoden Classification Tree als auch Logistische Regression an.
# Zeigen Sie die Trennschärfe Ihres Modells mit der ROC-Kurve.
# Quelle fuer Daten: https://rdrr.io/cran/kernlab/man/spam.html.

# deskriptive statistiken
data(spam)
attach(spam)
head(spam, n=5)
nrow(spam)
ncol(spam)
summary(spam)
cols <- colnames(spam)
cols2 <- paste(cols, collapse="+")

sum(spam$type=="nonspam")
sum(spam$type=="spam")

summary(spam[spam$type=="nonspam",])
summary(spam[spam$type=="spam",])

par(mfrow=c(2,2))
boxplot(spam$direct ~ spam$type, main="direct")
boxplot(spam$free ~ spam$type, main="free")
boxplot(spam$num000 ~ spam$type, main="num000")
boxplot(spam$credit ~ spam$type, main="credit")

# classification tree: recursive partitioning (rpart)
factor.type <- factor(spam$type)
is.factor(factor.type)

 # FALL 1: mit wahl von cp=0.0001 (ergibt großen tree mit vielen splits)
spam.rp1 <- rpart(factor.type ~ make+address+all+num3d+our+over+remove+
                      internet+order+mail+receive+will+people+
                      report+addresses+free+business+email+you+
                      credit+your+font+num000+money+hp+hpl+george+
                      num650+lab+labs+telnet+num857+data+num415+
                      num85+technology+num1999+parts+pm+direct+cs+
                      meeting+original+project+re+edu+table+conference+
                      charSemicolon+charRoundbracket+charSquarebracket+
                      charExclamation+charDollar+charHash+capitalAve+
                      capitalLong+capitalTotal, cp=0.0001)

summary(spam.rp1)
```

```
x11()
par(mfrow=c(1,1))
plot(spam.rp1)
text(spam.rp1, cex=0.6)

printcp(spam.rp1)
plotcp(spam.rp1)


# FALL 2: mit wahl von cp=0.003 (ergibt kleineren tree mit weniger splits)
spam.rp2 <- rpart(factor.type ~ make+address+all+num3d+our+over+remove+
                      internet+order+mail+receive+will+people+
                      report+addresses+free+business+email+you+
                      credit+your+font+num000+money+hp+hpl+george+
                      num650+lab+labs+telnet+num857+data+num415+
                      num85+technology+num1999+parts+pm+direct+cs+
                      meeting+original+project+re+edu+table+conference+
                      charSemicolon+charRoundbracket+charSquarebracket+
                      charExclamation+charDollar+charHash+capitalAve+
                      capitalLong+capitalTotal, cp=0.003)

summary(spam.rp2)

x11()
par(mfrow=c(1,1))
plot(spam.rp2)
text(spam.rp2, cex=0.6)

printcp(spam.rp2)
plotcp(spam.rp2)


# FALL 3: mit wahl von cp=0.007 (ergibt kleineren tree mit weniger splits)
spam.rp3 <- rpart(factor.type ~ make+address+all+num3d+our+over+remove+
                       internet+order+mail+receive+will+people+
                       report+addresses+free+business+email+you+
                       credit+your+font+num000+money+hp+hpl+george+
                       num650+lab+labs+telnet+num857+data+num415+
                       num85+technology+num1999+parts+pm+direct+cs+
                       meeting+original+project+re+edu+table+conference+
                       charSemicolon+charRoundbracket+charSquarebracket+
                       charExclamation+charDollar+charHash+capitalAve+
                       capitalLong+capitalTotal, cp=0.007)

summary(spam.rp3)

x11()
par(mfrow=c(1,1))
plot(spam.rp3)
text(spam.rp3, cex=0.6)

printcp(spam.rp3)
plotcp(spam.rp3)

# FALL 4: mit wahl von cp=0.05 (ergibt kleineren tree mit weniger splits)
spam.rp4 <- rpart(factor.type ~ make+address+all+num3d+our+over+remove+
                      internet+order+mail+receive+will+people+
                      report+addresses+free+business+email+you+
                      credit+your+font+num000+money+hp+hpl+george+
                      num650+lab+labs+telnet+num857+data+num415+
                      num85+technology+num1999+parts+pm+direct+cs+
                      meeting+original+project+re+edu+table+conference+
                      charSemicolon+charRoundbracket+charSquarebracket+
                      charExclamation+charDollar+charHash+capitalAve+
                      capitalLong+capitalTotal, cp=0.05)

summary(spam.rp4)
```

```
x11()
par(mfrow=c(1,1))
plot(spam.rp4)
text(spam.rp4, cex=0.6)

printcp(spam.rp4)
plotcp(spam.rp4)


# auto-pruning fuer fall 1:
spam.rp1.pruned <- prune(spam.rp1, cp=0.003)
x11()
plot(spam.rp1.pruned)
text(spam.rp1.pruned, cex=0.5)


# ROC-KURVEN, um trennschaerfe zu zeigen (fuer recursive partioning)
par(mfrow=c(2,2))

# ROC-kurve fuer fall 1:
# predict:
pred <- prediction(predict(spam.rp1, type = "prob")[, 2], factor.type)
# plot ROC-kurve:
plot(performance(pred, "tpr", "fpr"), col="mediumorchid3", lwd=2,
     main="Fall 1: cp=0.0001")
abline(0, 1, lty = 2, col="dodgerblue2")

# ROC-kurve fuer fall 2:
  # predict:
pred <- prediction(predict(spam.rp2, type = "prob")[, 2], factor.type)
  # plot ROC-kurve:
plot(performance(pred, "tpr", "fpr"), col="mediumorchid3", lwd=2,
     main="Fall 2: cp=0.003")
abline(0, 1, lty = 2, col="dodgerblue2")

# ROC-kurve fuer fall 3:
# predict:
pred <- prediction(predict(spam.rp3, type = "prob")[, 2], factor.type)
# plot ROC-kurve:
plot(performance(pred, "tpr", "fpr"), col="mediumorchid3", lwd=2,
     main="Fall 3: cp=0.007")
abline(0, 1, lty = 2, col="dodgerblue2")

# ROC-kurve fuer fall 4:
# predict:
pred <- prediction(predict(spam.rp4, type = "prob")[, 2], factor.type)
# plot ROC-kurve:
plot(performance(pred, "tpr", "fpr"), col="mediumorchid3", lwd=2,
     main="Fall 4: cp=0.007")
abline(0, 1, lty = 2, col="dodgerblue2")


# logistische regression

res.logit <- glm(factor.type ~ make+address+all+num3d+our+over+remove+
                   internet+order+mail+receive+will+people+
                   report+addresses+free+business+email+you+
                   credit+your+font+num000+money+hp+hpl+george+
                   num650+lab+labs+telnet+num857+data+num415+
                   num85+technology+num1999+parts+pm+direct+cs+
                   meeting+original+project+re+edu+table+conference+
                   charSemicolon+charRoundbracket+charSquarebracket+
                   charExclamation+charDollar+charHash+capitalAve+
                   capitalLong+capitalTotal,
                 family = binomial(link=logit))
res.logit
summary(res.logit)
```

```
res.probit <- glm(factor.type ~ make+address+all+num3d+our+over+remove+
                    internet+order+mail+receive+will+people+
                    report+addresses+free+business+email+you+
                    credit+your+font+num000+money+hp+hpl+george+
                    num650+lab+labs+telnet+num857+data+num415+
                    num85+technology+num1999+parts+pm+direct+cs+
                    meeting+original+project+re+edu+table+conference+
                    charSemicolon+charRoundbracket+charSquarebracket+
                    charExclamation+charDollar+charHash+capitalAve+
                    capitalLong+capitalTotal,
                  family = binomial(link=probit))
res.probit
summary(res.probit)

# ROC-kurven fuer logistische regression (logit):

# methode 1:
par(mfrow=c(1,1))
prob.pred = predict(res.logit,type=c("response"))
spam$prob.pred=prob.pred
roc.1 <- roc(factor.type ~ prob.pred, data=spam)
plot(roc.1, col="lightslateblue")

# methode 2:
modelfit <- glm(formula=factor.type ~ make+address+all+num3d+our+over+remove+
                  internet+order+mail+receive+will+people+
                  report+addresses+free+business+email+you+
                  credit+your+font+num000+money+hp+hpl+george+
                  num650+lab+labs+telnet+num857+data+num415+
                  num85+technology+num1999+parts+pm+direct+cs+
                  meeting+original+project+re+edu+table+conference+
                  charSemicolon+charRoundbracket+charSquarebracket+
                  charExclamation+charDollar+charHash+capitalAve+
                  capitalLong+capitalTotal,
                family=binomial(), data=spam, na.action=na.omit)
rocplot(modelfit)
```

Aufgabe 2:

Verwenden Sie den Datensatz http://archive.ics.uci.edu/ml/datasets/Bank+Marketing#. Zur Absicherung gegenüber Over-Fitting ziehen Sie eine Zufallsstichprobe von 70% der Daten. Entwickeln Sie damit einen Classification Tree und prüfen dessen Trennschärfe mittels der ROC-Analyse für Training- und Test-Daten (70% - 30%).

Die Daten beziehen sich auf Direktmarketingkampagnen einer Bank. Ziel ist es, vorherzusagen, ob ein Kunde ein Termingeld (*term deposit*) anlegen wird oder nicht (i.e. Variable `y`). Der Datensatz umfasst 45211 Beobachtungen mit je 17 Variablen. Die vorliegende Auswertung basiert auf dem Datensatz `bank-full.csv`. Variablen sind beispielsweise das Alter (`age`), die Art des Berufs (`job`), der Familienstand (`marital`) die Ausbildung (`education`), ob der Kunde bereits einen Kreditausfall hatte (`default`), wann der letzte Monat war, in dem der Kunde Kontakt mit der Bank hatte (`contact`), Ergebnis der vorigen Marketingkampagne (`poutcome`), und einige makroökonomische Variablen.
Von den 45211 Beobachtungen hatten 39922 in Bezug auf die abhängige Variable y die Ausprägung `no` und 5289 die Ausprägung `yes`.

Überblick über den Datensatz:

```
> head(bank.data, n=5)
  age           job marital education default balance housing loan contact day
1  58    management married  tertiary      no    2143     yes   no unknown   5
2  44    technician  single secondary      no      29     yes   no unknown   5
3  33 entrepreneur married secondary      no       2     yes  yes unknown   5
4  47  blue-collar married   unknown      no    1506     yes   no unknown   5
5  33       unknown  single   unknown      no       1      no   no unknown   5
  month duration campaign pdays previous poutcome  y
1   may      261        1    -1        0  unknown no
2   may      151        1    -1        0  unknown no
3   may       76        1    -1        0  unknown no
4   may       92        1    -1        0  unknown no
5   may      198        1    -1        0  unknown no
```

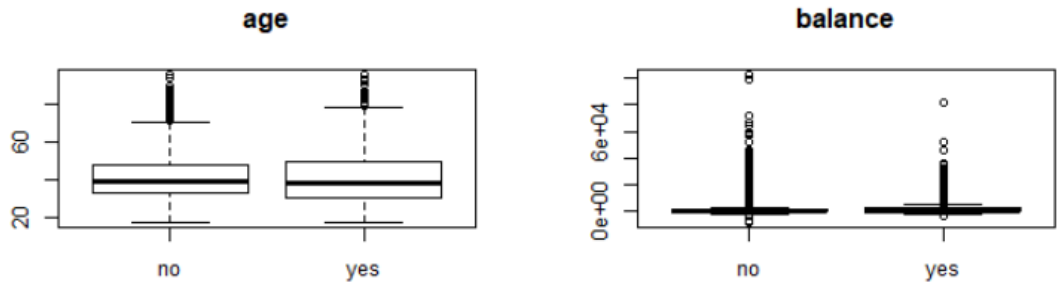
Deskriptive Statistiken:

```
> summary(bank.data[bank.data$y=="no",])
      age                job           marital          education     default
 Min.   :18.00   blue-collar:9024   divorced: 4585   primary  : 6260   no :39159
 1st Qu.:33.00   management :8157   married :24459   secondary:20752   yes:  763
 Median :39.00   technician :6757   single  :10878   tertiary :11305
 Mean   :40.84   admin.     :4540                    unknown  : 1605
 3rd Qu.:48.00   services   :3785
 Max.   :95.00   retired    :1748
                 (Other)    :5911
    balance       housing      loan            contact           day
 Min.   : -8019   no :16727   no :33162   cellular :24916   Min.   : 1.00
 1st Qu.:    58   yes:23195   yes: 6760   telephone: 2516   1st Qu.: 8.00
 Median :   417                          unknown  :12490   Median :16.00
 Mean   :  1304                                            Mean   :15.89
 3rd Qu.:  1345                                            3rd Qu.:21.00
 Max.   :102127                                            Max.   :31.00
```

```
     month          duration         campaign          pdays          previous
 may    :12841   Min.   :   0.0   Min.   : 1.000   Min.   : -1.00   Min.   :  0.0000
 jul    : 6268   1st Qu.:  95.0   1st Qu.: 1.000   1st Qu.: -1.00   1st Qu.:  0.0000
 aug    : 5559   Median : 164.0   Median : 2.000   Median : -1.00   Median :  0.0000
 jun    : 4795   Mean   : 221.2   Mean   : 2.846   Mean   : 36.42   Mean   :  0.5021
 nov    : 3567   3rd Qu.: 279.0   3rd Qu.: 3.000   3rd Qu.: -1.00   3rd Qu.:  0.0000
 apr    : 2355   Max.   :4918.0   Max.   :63.000   Max.   :871.00   Max.   :275.0000
 (Other): 4537
    poutcome       y
 failure: 4283   no :39922
 other  : 1533   yes:    0
 success:  533
 unknown:33573
```

```
> summary(bank.data[bank.data$y=="yes",])
      age                job          marital         education    default
 Min.   :18.00   management :1301   divorced: 622   primary  : 591   no :5237
 1st Qu.:31.00   technician : 840   married :2755   secondary:2450   yes:  52
 Median :38.00   blue-collar: 708   single  :1912   tertiary :1996
 Mean   :41.67   admin.     : 631                   unknown  : 252
 3rd Qu.:50.00   retired    : 516
 Max.   :95.00   services   : 369
                 (Other)    : 924
    balance       housing     loan           contact          day              month
 Min.   :-3058   no :3354   no :4805   cellular :4369   Min.   : 1.00   may    : 925
 1st Qu.:  210   yes:1935   yes: 484   telephone: 390   1st Qu.: 8.00   aug    : 688
 Median :  733              unknown  : 530   Median :15.00   jul    : 627
 Mean   : 1804                         Mean   :15.16   apr    : 577
 3rd Qu.: 2159                         3rd Qu.:22.00   jun    : 546
 Max.   :81204                         Max.   :31.00   feb    : 441
                                                       (Other):1485
    duration        campaign          pdays          previous        poutcome
 Min.   :   8.0   Min.   : 1.000   Min.   : -1.0   Min.   : 0.00   failure: 618
 1st Qu.: 244.0   1st Qu.: 1.000   1st Qu.: -1.0   1st Qu.: 0.00   other  : 307
 Median : 426.0   Median : 2.000   Median : -1.0   Median : 0.00   success: 978
 Mean   : 537.3   Mean   : 2.141   Mean   : 68.7   Mean   : 1.17   unknown:3386
 3rd Qu.: 725.0   3rd Qu.: 3.000   3rd Qu.: 98.0   3rd Qu.: 1.00
 Max.   :3881.0   Max.   :32.000   Max.   :854.0   Max.   :58.00

   y
 no :   0
 yes:5289
```

Boxplots für die Variablen `age`[1], `balance`[2], `duration`[3] und `pdays`[4] je nach Ausprägung von `y`:

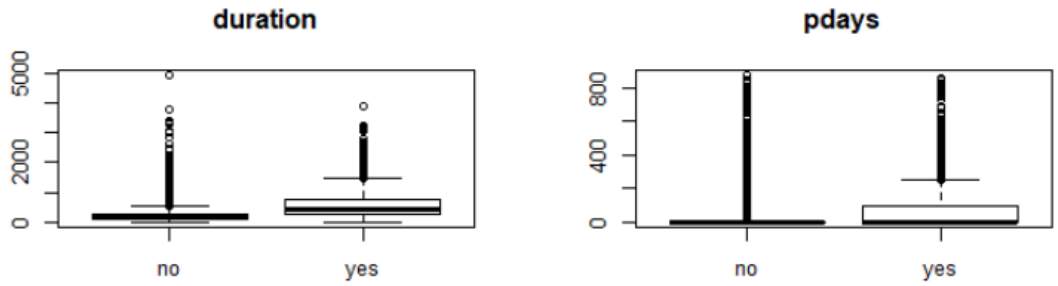


[1] `age`: Alter des Kunden in Jahren.

[2] `balance`: Durchschnittlicher jährlicher Kontostand des Kunden (in EUR).

[3] `duration`: Dauer des letzten Kontakts mit der Bank (in Sekunden).

[4] `pdays`: Anzahl von vergangenen Tagen, nachdem Kunde zuletzt von einer Kampagne informiert worden war.



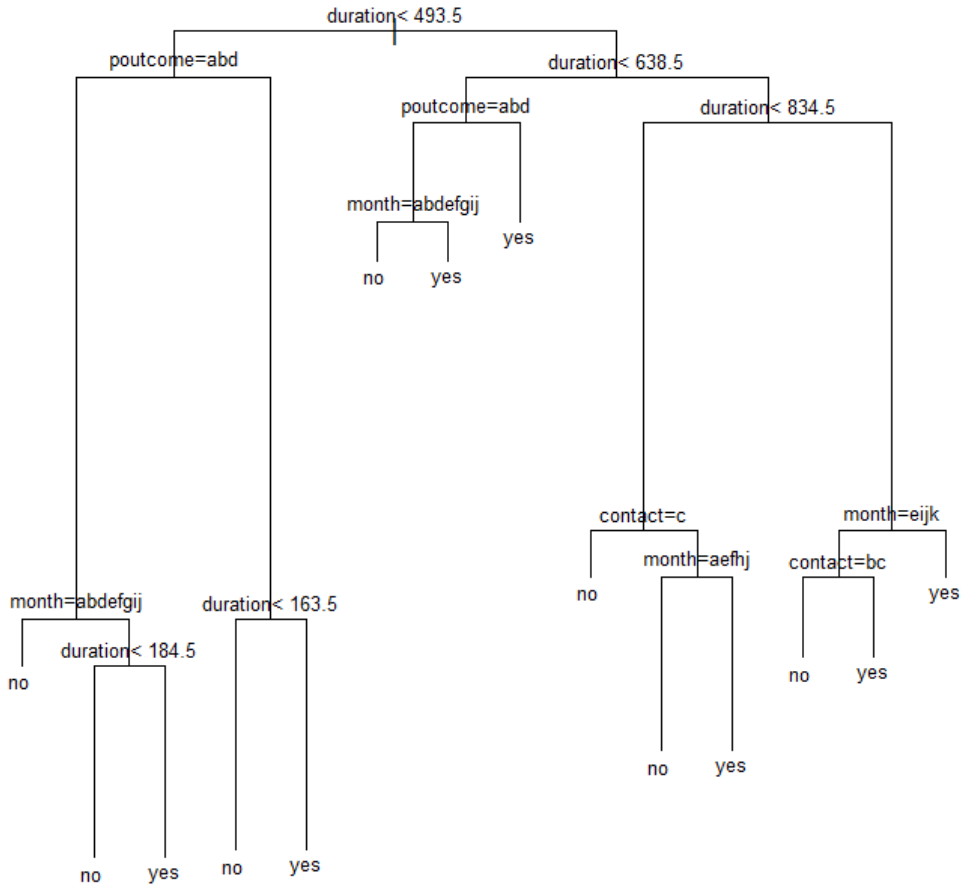
Nach einem deskriptiven Überblick wurde eine Zufallsstichprobe von 70% (der Beobachtungen) ohne Zurücklegen aus den Gesamtdaten gezogen. Diese 70% der Daten wurden als Trainingdaten verwendet, um den Classification Tree zu erzeugen. Die restlichen 30% der Daten wurden als Testdaten für die Vorhersage mittels Classification Tree verwendet.

```
# zufallsstichprobe ziehen (70% der daten)
training_size <- round(0.7*rows, digits=0)
random_indices <- sample(1:rows,training_size,replace=FALSE)
training.data <- bank.data[random_indices, ]
test.data <- bank.data[-random_indices, ]
```

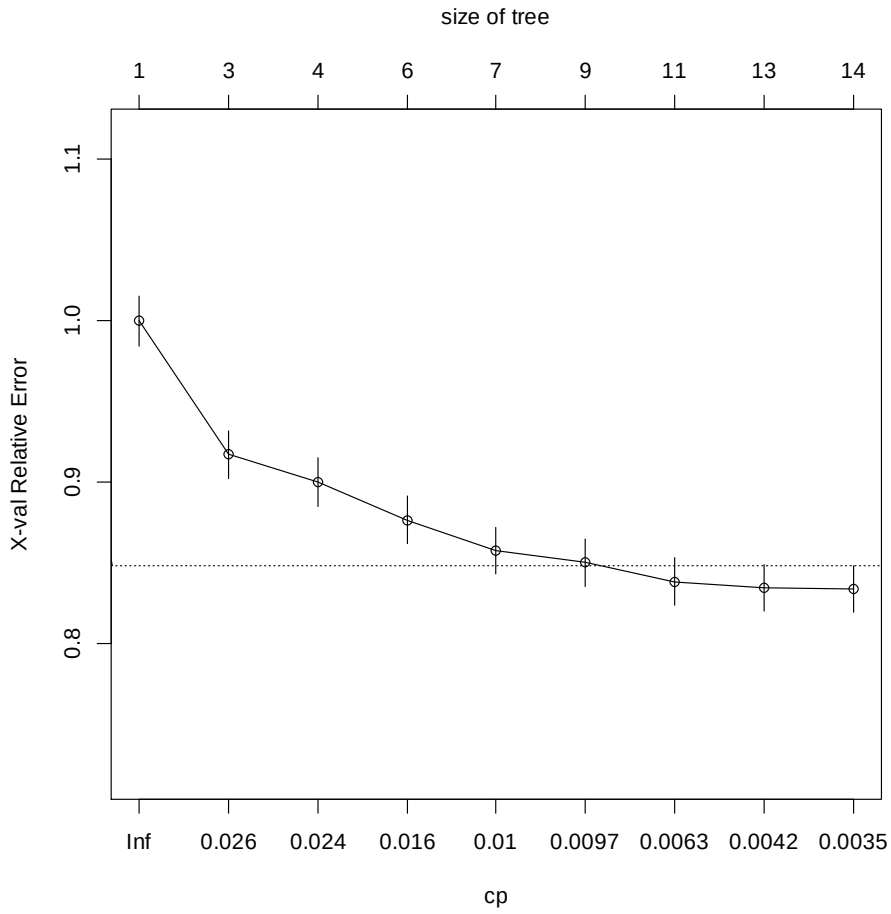
Basierend auf dem zufällig ermittelten Trainingsdatensatz wurde ein Classification Tree (hier genannt: `bank.rp`) für das volle Modell mittels Recursive Partitioning (`rpart`) mit einem cp-Wert von 0.003 erzeugt.

```
> summary(bank.rp)
Call:
rpart(formula = y ~ age + job + marital + education + default +
    balance + housing + loan + contact + day + month + duration +
    campaign + pdays + previous + poutcome, subset = splitting ==
    1, parms = list(split = "gini"), cp = 0.003)
  n= 31648

           CP nsplit rel error    xerror       xstd
1 0.028700906      0 1.0000000 1.0000000 0.01559013
2 0.024443834      2 0.9425982 0.9170558 0.01500986
3 0.023482560      3 0.9181544 0.9000275 0.01488613
4 0.010436693      5 0.8711892 0.8766822 0.01471379
5 0.010162043      6 0.8607525 0.8574567 0.01456945
6 0.009200769      8 0.8404285 0.8500412 0.01451317
7 0.004257072     10 0.8220269 0.8385059 0.01442496
8 0.004119747     12 0.8135128 0.8346608 0.01439537
9 0.003000000     13 0.8093930 0.8335622 0.01438690

Variable importance
duration poutcome    month  contact    pdays      day
      54       30       13        1        1        1
```

```
> bank.rp
n= 31648

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 31648 3641 no (0.88495324 0.11504676)
   2) duration< 493.5 27825 2039 no (0.92672058 0.07327942)
     4) poutcome=failure,other,unknown 26964 1504 no (0.94422193 0.05577807)
       8) month=apr,aug,feb,jan,jul,jun,may,nov 25999 1146 no (0.95592138 0.04407862) *
       9) month=dec,mar,oct,sep 965  358 no (0.62901554 0.37098446)
        18) duration< 184.5 519   98 no (0.81117534 0.18882466) *
        19) duration>=184.5 446  186 yes (0.41704036 0.58295964) *
     5) poutcome=success 861  326 yes (0.37862950 0.62137050)
      10) duration< 163.5 253   82 no (0.67588933 0.32411067) *
      11) duration>=163.5 608  155 yes (0.25493421 0.74506579) *
   3) duration>=493.5 3823 1602 no (0.58095736 0.41904264)
     6) duration< 638.5 1519  443 no (0.70836076 0.29163924)
      12) poutcome=failure,other,unknown 1447  388 no (0.73185902 0.26814098)
        24) month=apr,aug,feb,jan,jul,jun,may,nov 1384  349 no (0.74783237 0.25216763) *
        25) month=dec,mar,oct,sep 63   24 yes (0.38095238 0.61904762) *
      13) poutcome=success 72   17 yes (0.23611111 0.76388889) *
     7) duration>=638.5 2304 1145 yes (0.49696181 0.50303819)
      14) duration< 834.5 1087  465 no (0.57221711 0.42778289)
        28) contact=unknown 297   87 no (0.70707071 0.29292929) *
        29) contact=cellular,telephone 790  378 no (0.52151899 0.47848101)
          58) month=apr,jan,jul,mar,nov 407  153 no (0.62407862 0.37592138) *
          59) month=aug,dec,feb,jun,may,oct,sep 383  158 yes (0.41253264 0.58746736) *
      15) duration>=834.5 1217  523 yes (0.42974528 0.57025472)
        30) month=jan,may,nov,oct 516  253 yes (0.49031008 0.50968992)
          60) contact=telephone,unknown 227   98 no (0.56828194 0.43171806) *
          61) contact=cellular 289  124 yes (0.42906574 0.57093426) *
        31) month=apr,aug,dec,feb,jul,jun,mar,sep 701  270 yes (0.38516405 0.61483595) *
```



```
> printcp(bank.rp)

Classification tree:
rpart(formula = y ~ age + job + marital + education + default +
    balance + housing + loan + contact + day + month + duration +
    campaign + pdays + previous + poutcome, subset = splitting ==
    1, parms = list(split = "gini"), cp = 0.003)

Variables actually used in tree construction:
[1] contact  duration month    poutcome

Root node error: 3641/31648 = 0.11505

n= 31648

         CP nsplit rel error  xerror     xstd
1 0.0287009      0   1.00000 1.00000 0.015590
2 0.0244438      2   0.94260 0.91706 0.015010
3 0.0234826      3   0.91815 0.90003 0.014886
4 0.0104367      5   0.87119 0.87668 0.014714
5 0.0101620      6   0.86075 0.85746 0.014569
6 0.0092008      8   0.84043 0.85004 0.014513
7 0.0042571     10   0.82203 0.83851 0.014425
8 0.0041197     12   0.81351 0.83466 0.014395
9 0.0030000     13   0.80939 0.83356 0.014387
```



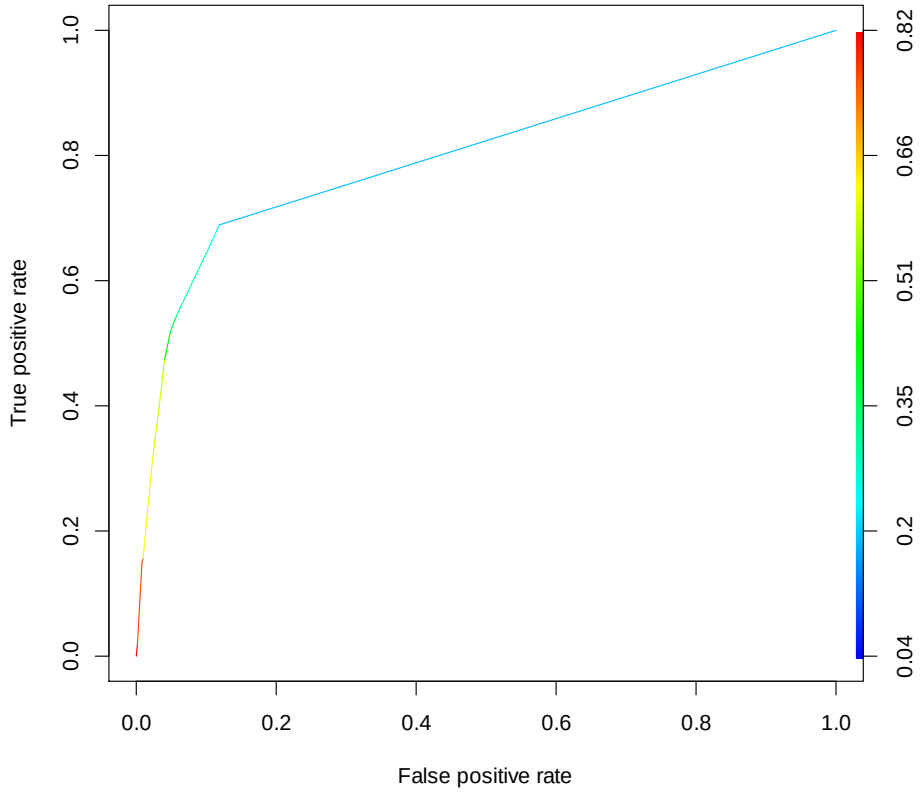
Danach wurde die Confusion-Matrix in den Trainings- und in den Testdaten jeweils getrennt berechnet.

Within sample: (basierend auf den Trainingsdaten)

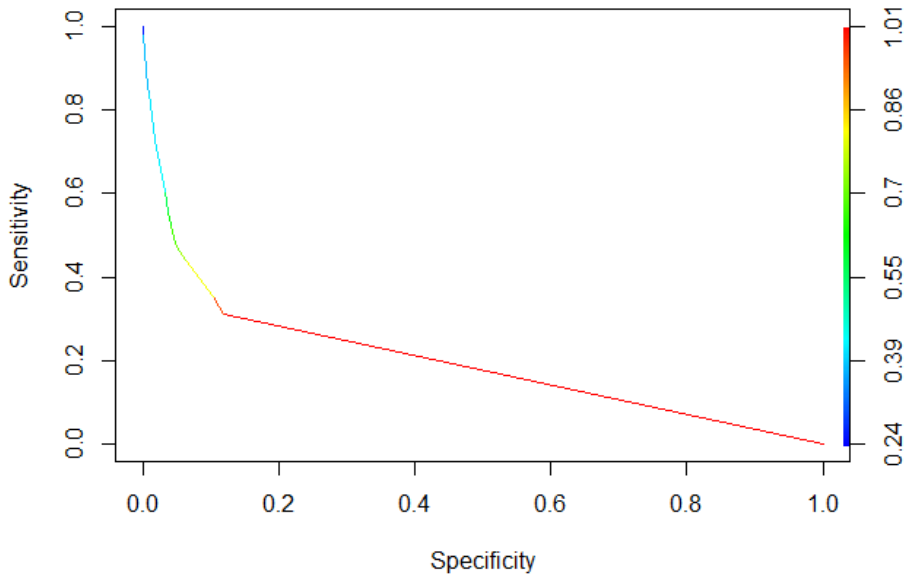
Die Sensitivity liegt bei 44.71% und die Specificity bei 96.67%.

```
> confusion(bank.preds.rpart, y[splitting==1])
[1] "Sensitivity:  44.71"
[1] "Specifity:  96.67"
         True: 0 True: 1
Pred: 0   27073    2013 29086
Pred: 1     934    1628  2562
          28007    3641 31648
```

ROC-Kurve basierend auf den Trainingsdaten:



Sensitivity-Specificity-Kurve basierend auf den Trainingsdaten:

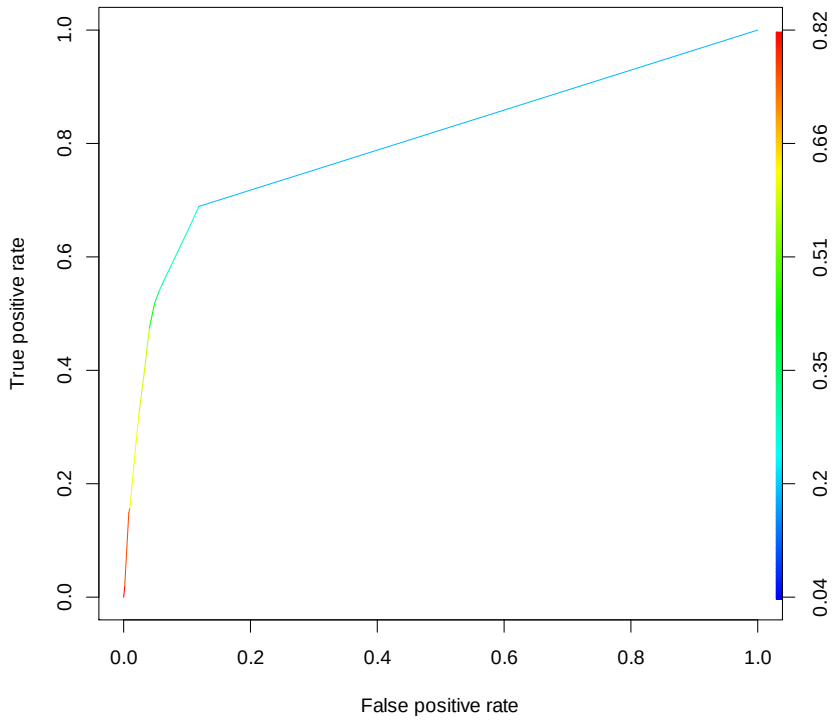


Out of sample: (basierend auf den zuvor ungesehenen Testdaten)

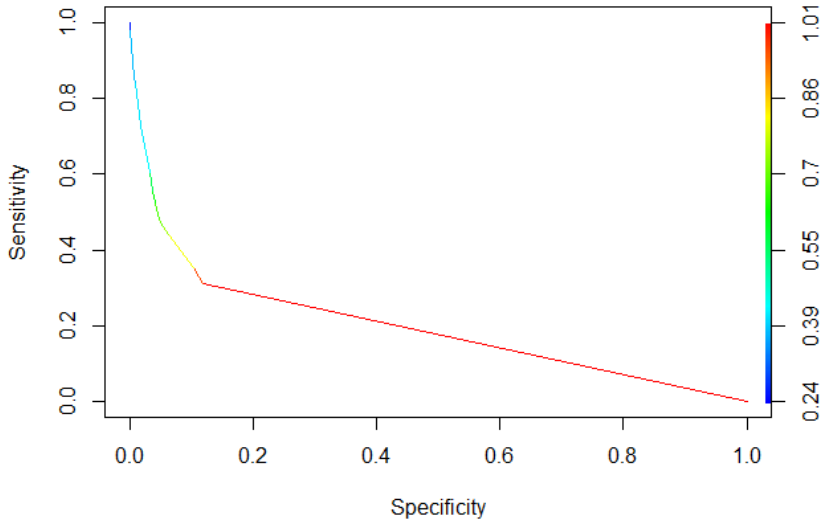
Die Sensitivity liegt bei 44.42% (also unter jenem Wert basierend auf den Trainingsdaten) und die Specificity bei 96.38% (ebenfalls niedriger als basierend auf den Trainingsdaten).

```
> confusion(bank.preds.rpart, y[splitting==2])
[1] "Sensitivity:  44.42"
[1] "Specifity:  96.38"
          True: 0 True: 1
Pred: 0    11484     916 12400
Pred: 1      431     732  1163
           11915    1648 13563
```

ROC-Kurve basierend auf den Testdaten:



Sensitivity-Specificity-Kurve basierend auf den Testdaten:



R-Code zu Aufgabe 2:

```
#*************************************************************************************************
# AUFGABE 2
#*************************************************************************************************
# Verwenden Sie den Datensatz http://archive.ics.uci.edu/ml/datasets/Bank+Marketing#.
# Zur Absicherung gegenüber Over-Fitting ziehen Sie eine Zufallsstichprobe von 70% der Daten.
# Entwickeln Sie damit einen Classification Tree und prüfen dessen Trennschärfe mittels der
# ROC-Analyse für Training- und Test-Daten (70% - 30%).
# anmerkung: y ... yes/no (has the client subscribed a term deposit?)

# daten einlesen
bank.data <- read.csv("bank-full.csv", sep=";")
attach(bank.data)

# deskriptive statistiken
rows <- nrow(bank.data)
cols <- ncol(bank.data)
head(bank.data, n=5)

summary(bank.data[bank.data$y=="no",])
summary(bank.data[bank.data$y=="yes",])

par(mfrow=c(2,2))
boxplot(age ~ y, main="age")
boxplot(balance ~ y, main="balance")
boxplot(duration ~ y, main="duration")
boxplot(pdays ~ y, main="pdays")

# zufallsstichprobe ziehen (70% der daten)
training_size <- round(0.7*rows, digits=0)
random_indices <- sample(1:rows,training_size,replace=FALSE)
training.data <- bank.data[random_indices, ]
test.data <- bank.data[-random_indices, ]

  # 1 ... training sample
  # 2 ... test sample
splitting <- rep(2,rows)
splitting[random_indices] <- 1 # 70% data is training sample

# classification tree (mit rpart)
is.factor(y) # y ist schon ein factor
cols <- colnames(bank.data)
cols2 <- paste(cols, collapse="+")

bank.rp <- rpart(y ~ age+job+marital+education+
                     default+balance+housing+loan+
                     contact+day+month+duration+
                     campaign+pdays+previous+poutcome,
                 subset=splitting==1,
                 parms=list(split="gini"),
                 cp=0.003)

summary(bank.rp)

x11()
par(mfrow=c(1,1))
plot(bank.rp)
text(bank.rp, cex=0.7)

printcp(bank.rp)
plotcp(bank.rp)
```

```
# ROC-kurve (pruefe trennschaerfe; mittels training (70%) - test (30%) split)

# within sample (for TRAINING data)
bank.probs.rpart <- predict(bank.rp, newdata=bank.data[splitting==1, ])
bank.preds.rpart <- predict(bank.rp, newdata=bank.data[splitting==1, ],
                            type="class")
confusion(bank.preds.rpart, y[splitting==1])

par(mfrow=c(1,2))
roc.plot(y[splitting==1], bank.probs.rpart[,2], legend=TRUE,
         leg.text="RPART",
         plot.thres=NULL, main="Training Sample")

par(mfrow=c(1,1))
library(caret)
library(ROCR)
roc_pred <- prediction(bank.probs.rpart[,2], training.data$y)
x11()
plot(performance(roc_pred, measure="tpr", x.measure="fpr"), colorize=TRUE)

# sensitivity / speficicity kurve
plot(performance(roc_pred, measure="sens", x.measure="spec"), colorize=TRUE)

# out of sample test (for test data)
bank.probs.rpart <- predict(bank.rp, newdata=bank.data[splitting==2, ])
bank.preds.rpart <- predict(bank.rp, newdata=bank.data[splitting==2, ],
                            type="class")
confusion(bank.preds.rpart, y[splitting==2])

x11()
roc.plot(as.numeric(as.character(y[splitting==2])), bank.probs.rpart[,2],
         legend=TRUE,
         leg.text="RPART",
         plot.thres=NULL, main="Test Sample")

par(mfrow=c(1,1))
library(caret)
library(ROCR)
roc_pred <- prediction(bank.probs.rpart[,2], test.data$y)
x11()
plot(performance(roc_pred, measure="tpr", x.measure="fpr"), colorize=TRUE)

# sensitivity / speficicity kurve
plot(performance(roc_pred, measure="sens", x.measure="spec"), colorize=TRUE)
```

Aufgabe 3:

Verwenden Sie den Datensatz https://stats.idre.ucla.edu/stat/data/binary.csv.
Zielvariable: Admission to graduate school admit =1 / don't admit =0.
Erklärende Variablen: GRE (Graduate Record Exam scores), GPA (grade point average) und Prestige der Undergraduate Institution (Faktor mit Werten 1-4),
Wenden Sie die Logistische Regression an und wählen Sie ein geeignetes Modell
(Prüfen Sie auch auf etwaige relevante Interaktionseffekte)
Zeigen Sie die Trennschärfe Ihres Modells mit der ROC-Kurve.

Der Datensatz umfasst 400 Beobachtungen für 3 erklärende Variablen (`gre`, `gpa`, `rank`) und eine Zielvariable (`admit`). Die Zielvariable gibt an, ob ein Bewerber in die Graduate School aufgenommen wurde (1) oder nicht (0).

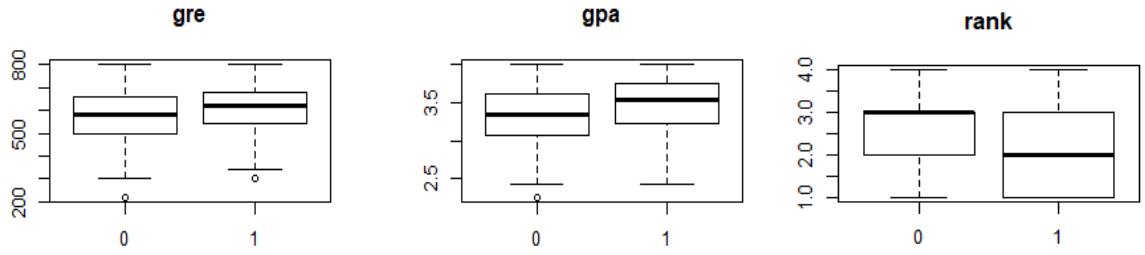
Überblick über den Datensatz:

```
> head(grad.data, n=10)
   admit gre  gpa rank
1      0 380 3.61    3
2      1 660 3.67    3
3      1 800 4.00    1
4      1 640 3.19    4
5      0 520 2.93    4
6      1 760 3.00    2
7      1 560 2.98    1
8      0 400 3.08    2
9      1 540 3.39    3
10     0 700 3.92    2
```

Deskriptive Statistiken:

```
> summary(grad.data[grad.data$admit==0,])
     admit        gre             gpa             rank      
 Min.   :0   Min.   :220.0   Min.   :2.260   Min.   :1.000  
 1st Qu.:0   1st Qu.:500.0   1st Qu.:3.080   1st Qu.:2.000  
 Median :0   Median :580.0   Median :3.340   Median :3.000  
 Mean   :0   Mean   :573.2   Mean   :3.344   Mean   :2.641  
 3rd Qu.:0   3rd Qu.:660.0   3rd Qu.:3.610   3rd Qu.:3.000  
 Max.   :0   Max.   :800.0   Max.   :4.000   Max.   :4.000  
> summary(grad.data[grad.data$admit==1,])
     admit        gre             gpa             rank      
 Min.   :1   Min.   :300.0   Min.   :2.420   Min.   :1.00  
 1st Qu.:1   1st Qu.:540.0   1st Qu.:3.220   1st Qu.:1.00  
 Median :1   Median :620.0   Median :3.540   Median :2.00  
 Mean   :1   Mean   :618.9   Mean   :3.489   Mean   :2.15  
 3rd Qu.:1   3rd Qu.:680.0   3rd Qu.:3.755   3rd Qu.:3.00  
 Max.   :1   Max.   :800.0   Max.   :4.000   Max.   :4.00  
```

Boxplots der erklärenden Variablen je nach Ausprägung der Zielvariable `admit`:



Zur Erklärung der Zielvariable durch die weiteren 3 Variablen wurde eine **logistische Regression** durchgeführt, wobei 4 verschiedene Modelle betrachtet wurden. Einerseits wurde ein additives Modell, das alle 3 Variablen beinhaltet, gerechnet, andererseits ein Modell mit Interaktionen aller Variablen und 2 Modelle mit nur je einer Interaktion. Für Modell 4, ein Modell mit einer Interaktion, wurde sodann die ROC-Kurve dargestellt. Die Fläche unter der ROC-Kurve beträgt ca. 69%.

Modell 1 (Additives Modell mit allen erklärenden Variablen):

```
> summary(res.logit.1)

Call:
glm(formula = factor.admit ~ gre + gpa + rank, family = binomial(link = logit))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5802  -0.8848  -0.6382   1.1575   2.1732

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.449548   1.132846  -3.045  0.00233 **
gre          0.002294   0.001092   2.101  0.03564 *
gpa          0.777014   0.327484   2.373  0.01766 *
rank        -0.560031   0.127137  -4.405 1.06e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 459.44  on 396  degrees of freedom
AIC: 467.44

Number of Fisher Scoring iterations: 4
```

Modell 2 (Modell mit Interaktionen zwischen allen Variablen):

```
> summary(res.logit.2)

Call:
glm(formula = factor.admit ~ gre * gpa * rank, family = binomial(link = logit))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5070  -0.8927  -0.6241   1.1409   2.2592

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.508e+01  1.641e+01  -0.919    0.358
gre           1.882e-02  2.657e-02   0.708    0.479
gpa           4.361e+00  4.845e+00   0.900    0.368
rank          4.728e-02  6.607e+00   0.007    0.994
gre:gpa      -5.101e-03  7.747e-03  -0.659    0.510
gre:rank      2.152e-04  1.088e-02   0.020    0.984
gpa:rank     -2.414e-01  1.934e+00  -0.125    0.901
gre:gpa:rank  4.184e-05  3.147e-03   0.013    0.989

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 456.25  on 392  degrees of freedom
AIC: 472.25

Number of Fisher Scoring iterations: 5
```

Modell 3 (Modell mit Interaktion zwischen `gpa` und `rank`):

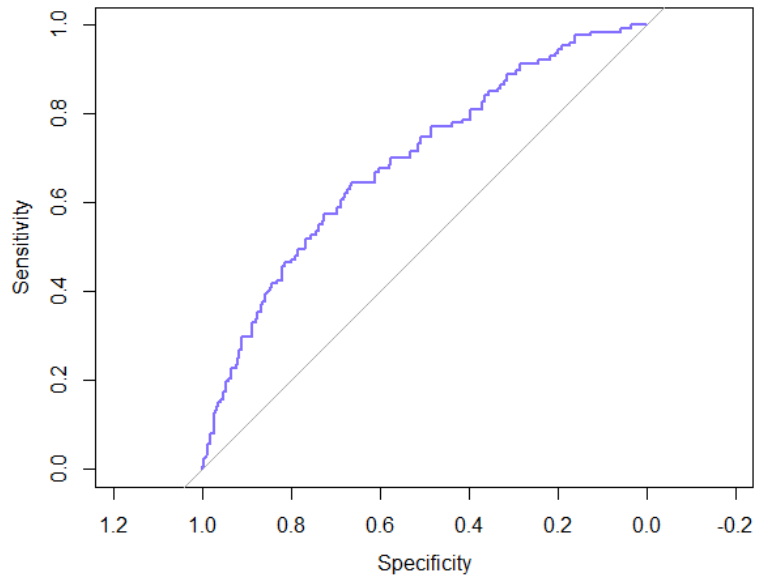
```
> summary(res.logit.3)

Call:
glm(formula = factor.admit ~ gre + gpa * rank, family = binomial(link = logit))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6161  -0.8744  -0.6357   1.1516   2.1563

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.344688   2.967727  -1.464   0.1432
gre          0.002300   0.001093   2.104   0.0354 *
gpa          1.036651   0.860434   1.205   0.2283
rank        -0.167427   1.204479  -0.139   0.8894
gpa:rank    -0.114225   0.348850  -0.327   0.7433
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 459.33  on 395  degrees of freedom
AIC: 469.33

Number of Fisher Scoring iterations: 4
```

Modell 4 (Modell mit Interaktion zwischen `gre` und `rank`):

```
> summary(res.logit.4)

Call:
glm(formula = factor.admit ~ gre + gpa + rank + gre:rank, family = binomial(link = logit))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5780  -0.8848  -0.6379   1.1576   2.1758

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.423e+00  1.915e+00  -1.788   0.0738 .
gre          2.250e-03  2.789e-03   0.807   0.4198
gpa          7.771e-01  3.275e-01   2.373   0.0177 *
rank        -5.714e-01  6.767e-01  -0.844   0.3984
gre:rank     1.889e-05  1.101e-03   0.017   0.9863
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 499.98  on 399  degrees of freedom
Residual deviance: 459.44  on 395  degrees of freedom
AIC: 469.44

Number of Fisher Scoring iterations: 4
```

ROC-Kurve (für Modell 4):

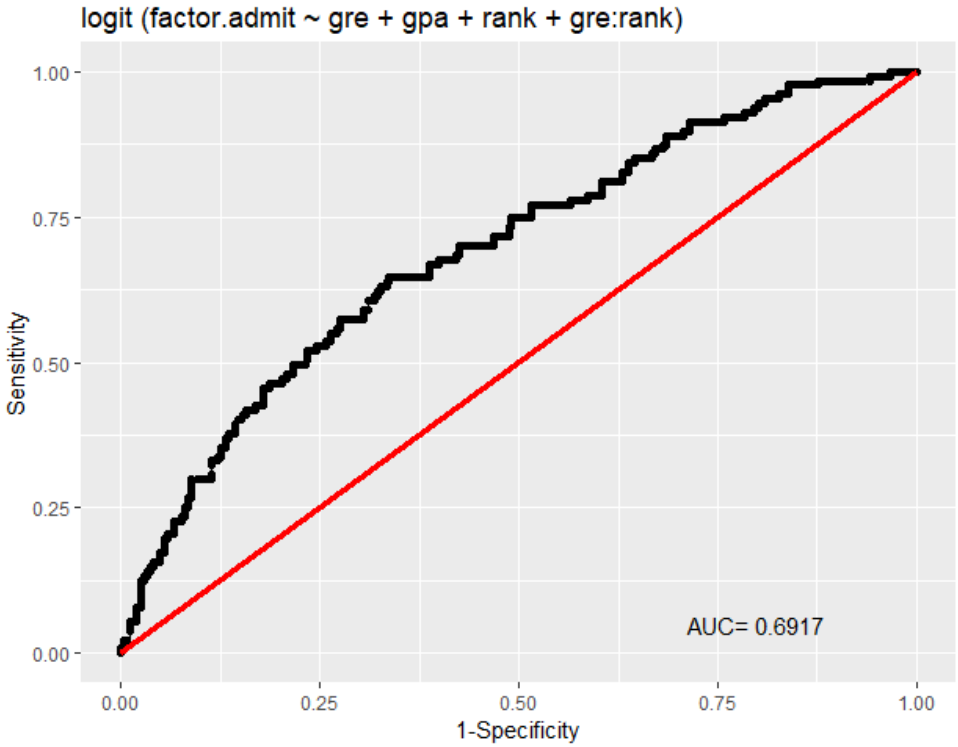
Methode 1:



Call:
roc.formula(formula = factor.admit ~ prob.pred, data = grad.data)

Data: prob.pred in 273 controls (factor.admit 0) < 127 cases (factor.admit 1).
Area under the curve: 0.6917

Methode 2:



R-Code zu Aufgabe 3:

```
#*************************************************************************************************
# AUFGABE 3
#*************************************************************************************************
# Verwenden Sie den Datensatz https://stats.idre.ucla.edu/stat/data/binary.csv.
# Zielvariable: Admission to graduate school admit =1 / don't admit =0.
# Erklärende Variablen: GRE (Graduate Record Exam scores), GPA (grade point average) und
#                       Prestige der Undergraduate Institution (Faktor mit Werten 1-4).
# Wenden Sie die Logistische Regression an und wählen Sie ein geeignetes Modell.
# (Prüfen Sie auch auf etwaige relevante Interaktionseffekte.)
# Zeigen Sie die Trennschärfe Ihres Modells mit der ROC-Kurve.

# daten einlesen
grad.data <- read.csv("binary.csv", sep=",")
attach(grad.data)

# deskriptive statistiken
rows <- nrow(grad.data)
cols <- ncol(grad.data)
head(grad.data, n=10)

summary(grad.data[grad.data$admit==0,])
summary(grad.data[grad.data$admit==1,])

par(mfrow=c(2,2))
boxplot(gre ~ admit, main="gre")
boxplot(gpa ~ admit, main="gpa")
boxplot(rank ~ admit, main="rank")

factor.admit <- factor(admit)

# logistische regression & modellwahl
# modell 1: additives modell
res.logit.1 <- glm(factor.admit ~ gre + gpa + rank,
                   family = binomial(link=logit))
res.logit.1
summary(res.logit.1)

# modell 2: mit interaktionen
res.logit.2 <- glm(factor.admit ~ gre*gpa*rank,
                   family = binomial(link=logit))
res.logit.2
summary(res.logit.2)

# modell 3: mit 1 interaktion
res.logit.3 <- glm(factor.admit ~ gre + gpa*rank,
                     family = binomial(link=logit))
res.logit.3
summary(res.logit.3)

# modell 4: mit 1 interaktion
res.logit.4 <- glm(factor.admit ~ gre + gpa + rank + gre:rank,
                     family = binomial(link=logit))
res.logit.4
summary(res.logit.4)
```

```
# ROC-kurve
# (beispielhaft fuer modell 4)

# methode 1:
par(mfrow=c(1,1))
prob.pred = predict(res.logit.4,type=c("response"))
grad.data$prob.pred=prob.pred
roc.1 <- roc(factor.admit ~ prob.pred, data=grad.data)
plot(roc.1, col="lightslateblue")

# methode 2:
modelfit <- glm(formula=factor.admit ~ gre + gpa + rank,
                family=binomial(), data=grad.data,
                na.action=na.omit)
rocplot(modelfit)
```

Literaturquellen:

- Folien und R-Codes zu den bisher vorgetragenen Kapiteln aus UK Erweiterungen des linearen Modells (Prof. Marcus Hudec).
- https://stats.stackexchange.com/questions/49416/decision-tree-model-evaluation-for-training-set-vs-testing-set-in-r
- https://rdrr.io/cran/kernlab/man/spam.html
- http://archive.ics.uci.edu/ml/datasets/Bank+Marketing#
- https://stats.idre.ucla.edu/stat/data/binary.csv