Beispielanalyse zum Prognose-Wettbewerb

Sebastian Sauer

Table of Contents

# Beschreibung der Aufgabe

Im Datensatz tips soll die Zielvariable tip im *Test*-Datensatz vorhergesagt werden. Dazu können alle übrigen Variablen im Trainings-Datensatz als Prädiktoren herangezogen werden.

# Pakete laden

library(mosaic)

# Daten laden

Den Train-Datensatz haben Sie vorliegen. Den Test-Datensatz nutzen Sie zur Vorhersage. Die “Lösung” – die voherzusagenden Werte – sind nur der Lehrperson bekannt.

tips\_train <- read.csv2("../data/tips\_train.csv")  
tips\_test <- read.csv2("../data/tips\_test.csv")

# Unbekannte Variable auf NA setzen

Das Trinkgeld (tip) ist vorherzusagen, aber unbekannt. Löschen wir also die Werte dieser Variablen. Die harte Wirklichkeit …

Zuerst ein Backup:

tips\_test\_mit\_Loesung <- tips\_test

Dann das Löschen:

tips\_test$tip <- NA

Keine Sorge: Dieser Schritt wird von der Lehrperson erledigt. Sie bekommen schon den “richtigen” Test-Datensatz (wo die vorherzusagende Variable auf NA gesetzt ist).

# Datenvorverarbeitung

Z.B. Analyse nach Extremwerten, Datentransformation; hier nicht ausgeführt.

# Ein Blick in die Daten

head(tips\_train)

## total\_bill tip sex smoker day time size  
## 1 16.99 1.01 Female No Sun Dinner 2  
## 2 21.01 3.50 Male No Sun Dinner 3  
## 3 24.59 3.61 Female No Sun Dinner 4  
## 4 26.88 3.12 Male No Sun Dinner 4  
## 5 10.27 1.71 Male No Sun Dinner 2  
## 6 35.26 5.00 Female No Sun Dinner 4

head(tips\_test)

## total\_bill tip sex smoker day time size  
## 1 10.34 NA Male No Sun Dinner 3  
## 2 23.68 NA Male No Sun Dinner 2  
## 3 25.29 NA Male No Sun Dinner 4  
## 4 8.77 NA Male No Sun Dinner 2  
## 5 15.04 NA Male No Sun Dinner 2  
## 6 14.78 NA Male No Sun Dinner 2

# Modellierung im Trainings-Datensatz (Phase 1)

## Modell 1

modell1 <- lm(tip ~ smoker, data = tips\_train)  
summary(modell1)

##   
## Call:  
## lm(formula = tip ~ smoker, data = tips\_train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.8709 -0.9709 -0.0578 0.5674 3.9922   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.7378 0.1357 20.173 <2e-16 \*\*\*  
## smokerYes 0.2331 0.2210 1.055 0.294   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.183 on 120 degrees of freedom  
## Multiple R-squared: 0.009185, Adjusted R-squared: 0.0009284   
## F-statistic: 1.112 on 1 and 120 DF, p-value: 0.2937

Hm, das ist noch nicht so gut…

## Modell 2

modell2 <- lm(tip ~ smoker + size, data = tips\_train)  
summary(modell2)

##   
## Call:  
## lm(formula = tip ~ smoker + size, data = tips\_train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.9728 -0.7046 -0.1896 0.6688 2.9508   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.7349 0.2610 2.816 0.00569 \*\*   
## smokerYes 0.4325 0.1772 2.441 0.01613 \*   
## size 0.7611 0.0903 8.429 9.42e-14 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9401 on 119 degrees of freedom  
## Multiple R-squared: 0.3796, Adjusted R-squared: 0.3692   
## F-statistic: 36.4 on 2 and 119 DF, p-value: 4.621e-13

Schon viel besser!

## Modell 3

modell3 <- lm(tip ~ smoker + size + sex, data = tips\_train)  
summary(modell3)

##   
## Call:  
## lm(formula = tip ~ smoker + size + sex, data = tips\_train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.9570 -0.7286 -0.2061 0.6683 2.9823   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.76475 0.26684 2.866 0.00493 \*\*   
## smokerYes 0.43850 0.17802 2.463 0.01521 \*   
## size 0.77078 0.09213 8.366 1.38e-13 \*\*\*  
## sexMale -0.10017 0.17530 -0.571 0.56880   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.9428 on 118 degrees of freedom  
## Multiple R-squared: 0.3813, Adjusted R-squared: 0.3656   
## F-statistic: 24.24 on 3 and 118 DF, p-value: 2.72e-12

Scheint nicht mehr besser zu werden ?!

Oder probieren wir doch noch ein anderes Modell …

# Modell 4 - mit Log-Transformation

modell4 <- lm(log(tip) ~ smoker + size + sex, data = tips\_train)  
summary(modell4)

##   
## Call:  
## lm(formula = log(tip) ~ smoker + size + sex, data = tips\_train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.86407 -0.26624 -0.01783 0.26655 0.67427   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.28936 0.09589 3.018 0.00312 \*\*   
## smokerYes 0.17542 0.06397 2.742 0.00705 \*\*   
## size 0.24730 0.03311 7.470 1.52e-11 \*\*\*  
## sexMale -0.04625 0.06299 -0.734 0.46428   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3388 on 118 degrees of freedom  
## Multiple R-squared: 0.3341, Adjusted R-squared: 0.3171   
## F-statistic: 19.73 on 3 and 118 DF, p-value: 1.958e-10

*Tipp:* Es gibt auch Funktionen zur automatischen Modellwahl.

## Achtung bei transformierten AV!

Haben Sie in Ihrem Modell die AV (Ziel-Variable) transformiert (z.B. logarithmiert), so müssen Sie *trotzdem* die untransformierten Vorhersagen einreichen!

Vorhersagen in der Log-Skala:

modell4\_predictions\_log <- predict(modell4, newdata = tips\_test)  
head(modell4\_predictions\_log)

## 1 2 3 4 5 6   
## 0.9850096 0.7377118 1.2323074 0.7377118 0.7377118 0.7377118

Vorhersagen rücktransformiert in die “normale” Skalierung – allerdings muss hier ein Korrekturterm ergänzt werden, hier z. B. unter der Annahme einer Normalverteilung der Residuen. Alternativ könnte auch zu Beginn ein generalisiertes lineares Modell mit Linkfunktion (glm()) verwendet werden.

modell4\_predictions <- exp(modell4\_predictions\_log)  
modell4\_predictions <- modell4\_predictions \* exp((sum(modell4$residuals^2)/modell4$df.residual)/2)  
  
head(modell4\_predictions)

## 1 2 3 4 5 6   
## 2.836004 2.214659 3.631675 2.214659 2.214659 2.214659

*Hinweis*: Die Exponentialfunktion ist die Umkehrfunktion zur Logarithmus-Funktion.

Einzureichen ist also modell4\_predictions – nicht die logarithmierten Vorhersagen!

# Vorhersage der Zielvariablen im Test-Datendatz (Phase 2)

Wir nehmen unser bestes Modell, um die Zielvariable im Test-Datensatz vorherzusagen.

modell3\_predictions <- predict(modell3, newdata = tips\_test)

*Diese* Daten reichen Sie dann ein.

# CSV-Datei mit den vorgesagten Werten erstellen

Der Name der CSV-Datei sollte das Format aufweisen Vorhersage\_IhrName.csv.

write.csv2(modell3\_predictions, "Vorhersage\_RudiRaetsel.csv")

Diese CSV-Datei Vorhersage\_RudiRaetsel.csv reichen Sie ein!

# Güte der Vorhersage bemessen (Phase 3, von der Lehrperson durchgeführt)

Diese Phase wird vom Dozenten durchgeführt. Sie müssen diese Phase nicht durchführen.

## Funktionen zur Berechnung der Modellgüte

:

r2 <- function(predicted, observed) {  
   
 rss <- sum((predicted - observed) ^ 2) ## residual sum of squares  
 tss <- sum((observed - mean(observed)) ^ 2) ## total sum of squares  
 rsq <- 1 - rss/tss  
   
 rsq <- c(rsq = rsq)  
  
 return(rsq)  
  
}

:

mae <- function(predicted, observed)  
{  
 error <- predicted - observed  
 mae <- mean(abs(error))  
 mae <- c(mae = mae)  
   
 return(mae)  
}

Für das Modell 3:

modell3\_r2 <- r2(modell3\_predictions, tips\_test\_mit\_Loesung$tip)  
modell3\_r2

## rsq   
## 0.1080589

modell3\_mae <- mae(modell3\_predictions, tips\_test\_mit\_Loesung$tip)  
modell3\_mae

## mae   
## 1.037093

Für das Modell 4:

modell4\_r2 <- r2(modell4\_predictions, tips\_test\_mit\_Loesung$tip)  
modell4\_r2

## rsq   
## 0.0645858

modell4\_mae <- mae(modell4\_predictions, tips\_test\_mit\_Loesung$tip)  
modell4\_mae

## mae   
## 1.061948

Hm, die Vorhersagequalität war noch nicht so gut. Vielleicht hätten wir den Datensatz noch besser aufbereiten sollen? Oder mehr/andere Prädiktoren in das Modell aufnehmen sollen? Oder …