Restaurant Data with Consumer Rating

Roman Mühlfeldner and Ana Stanusic

1 Data Science Project: Restaurant Data with Customer Rating

1.1 Libraries

```
library(plyr)
library(gplots)
```

```
##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(RColorBrewer)
library(ggplot2)
library(ggmosaic)
library(ggrepel)
library(leaflet)
library("caret")
```

```
## Loading required package: lattice
```

```
library("class")
library("e1071")
library("rpart")
library("randomForest")
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(nnet)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v tibble  2.1.3     v purrr   0.3.2
## v tidyr   1.0.0     v dplyr   0.8.3
## v readr   1.3.1     v stringr 1.4.0
## v tibble  2.1.3     v forcats 0.4.0

## -- Conflicts ---------------------------------------------------------------------------------------
## x dplyr::arrange()       masks plyr::arrange()
## x dplyr::combine()       masks randomForest::combine()
## x purrr::compact()       masks plyr::compact()
## x dplyr::count()         masks plyr::count()
## x dplyr::failwith()      masks plyr::failwith()
## x dplyr::filter()        masks stats::filter()
## x dplyr::id()            masks plyr::id()
## x dplyr::lag()           masks stats::lag()
## x purrr::lift()          masks caret::lift()
## x randomForest::margin() masks ggplot2::margin()
## x dplyr::mutate()        masks plyr::mutate()
## x dplyr::rename()        masks plyr::rename()
## x dplyr::summarise()     masks plyr::summarise()
## x dplyr::summarize()     masks plyr::summarize()
```

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

1.2 Aufgabenstellung:

- Datenaufbereitung [10%]
- Explorative Datenanalyse, speziell Visualisierung [20%]
- Modellierung (Klassifikation oder Regression) mit zumindest 3 Methoden, inkl. Parameter Tuning und Benchmarking [30%]
- Deployment des besten Modells mittels Web Service [10%]
- Kurzpräsentation des Projekts/der Ergebnisse mittels Dashboard [10%]
- Extra-Feature - zB neue Methoden, interaktive Visualisierung [20%]
- Dokumentation mittels Notebook

1.3 Datenaufbereitung

Daten einlesen:

```
accept <- read.csv('./data/chefmozaccepts.csv')
cuisine <- read.csv('./data/chefmozcuisine.csv')
hours <- read.csv('./data/chefmozhours4.csv')
parking <- read.csv('./data/chefmozparking.csv')
geoplaces <- read.csv('./data/geoplaces2.csv', na.strings = "?")
rating <- read.csv('./data/rating_final.csv')
usercuisine <- read.csv('./data/usercuisine.csv')
userpayment <- read.csv('./data/userpayment.csv')
userprofile <- read.csv('./data/userprofile.csv', na.strings = "?")
```

Die Daten bestehen aus verschiedenen Datensätzen, die von Restaurants in Mexiko stammen. Grundsätzlich sind die Daten in drei Gruppen zu unterteilen: 1. Restaurant-Sicht (inklusive den Geo-Daten) 2. Kunden-Sicht 3. Rating vom Kunden des Restaurant

1.4 Datenanalyse

1.4.1 Restaurant-Daten

```
head(accept)
```

```
##   placeID            Rpayment
## 1  135110                cash
## 2  135110                VISA
## 3  135110 MasterCard-Eurocard
## 4  135110    American_Express
## 5  135110    bank_debit_cards
## 6  135109                cash
```

```
dim(accept)
```

```
## [1] 1314    2
```

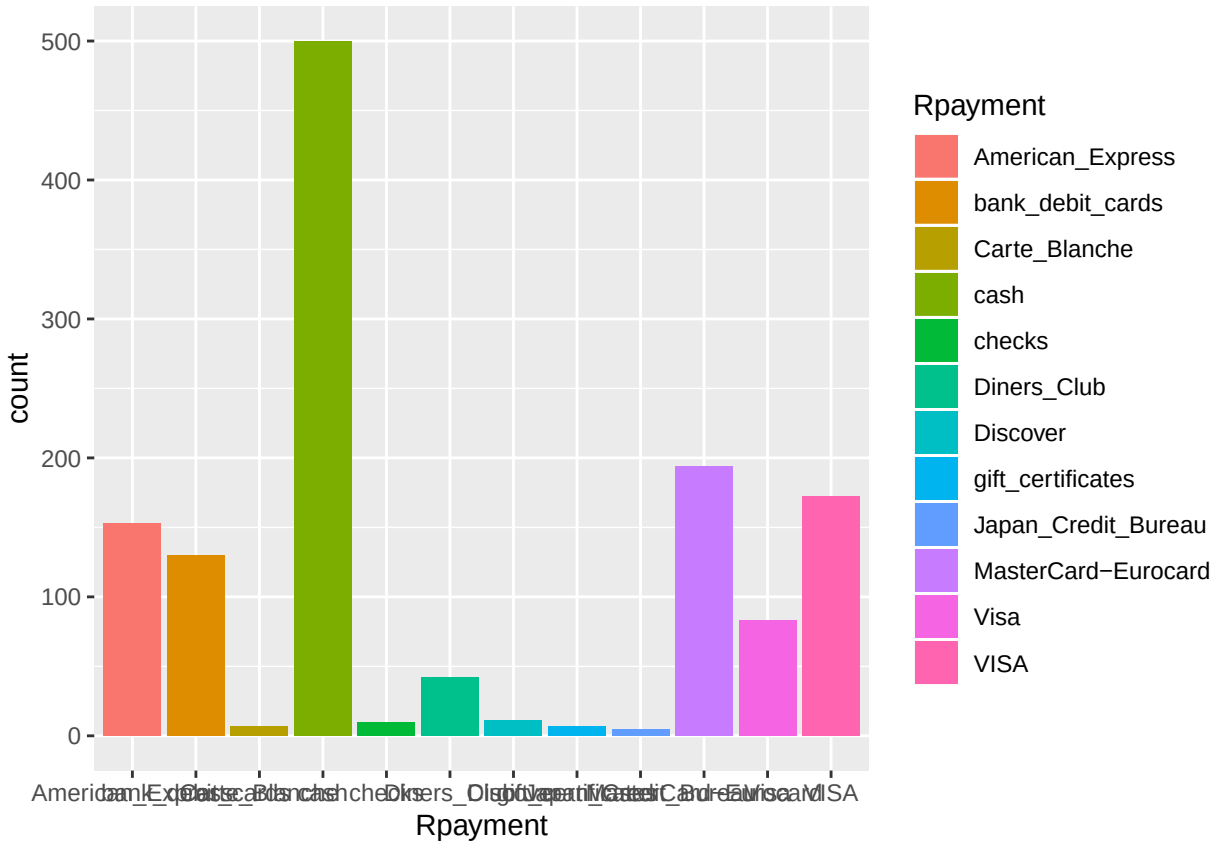
```
summary(accept)
```

```
##     placeID                     Rpayment
##  Min.   :132002   cash                :500
##  1st Qu.:132580   MasterCard-Eurocard:194
##  Median :132788   VISA                :172
##  Mean   :133219   American_Express    :153
##  3rd Qu.:133036   bank_debit_cards    :130
##  Max.   :135110   Visa                : 83
##                   (Other)             : 82
```

```
levels(accept$Rpayment)
```

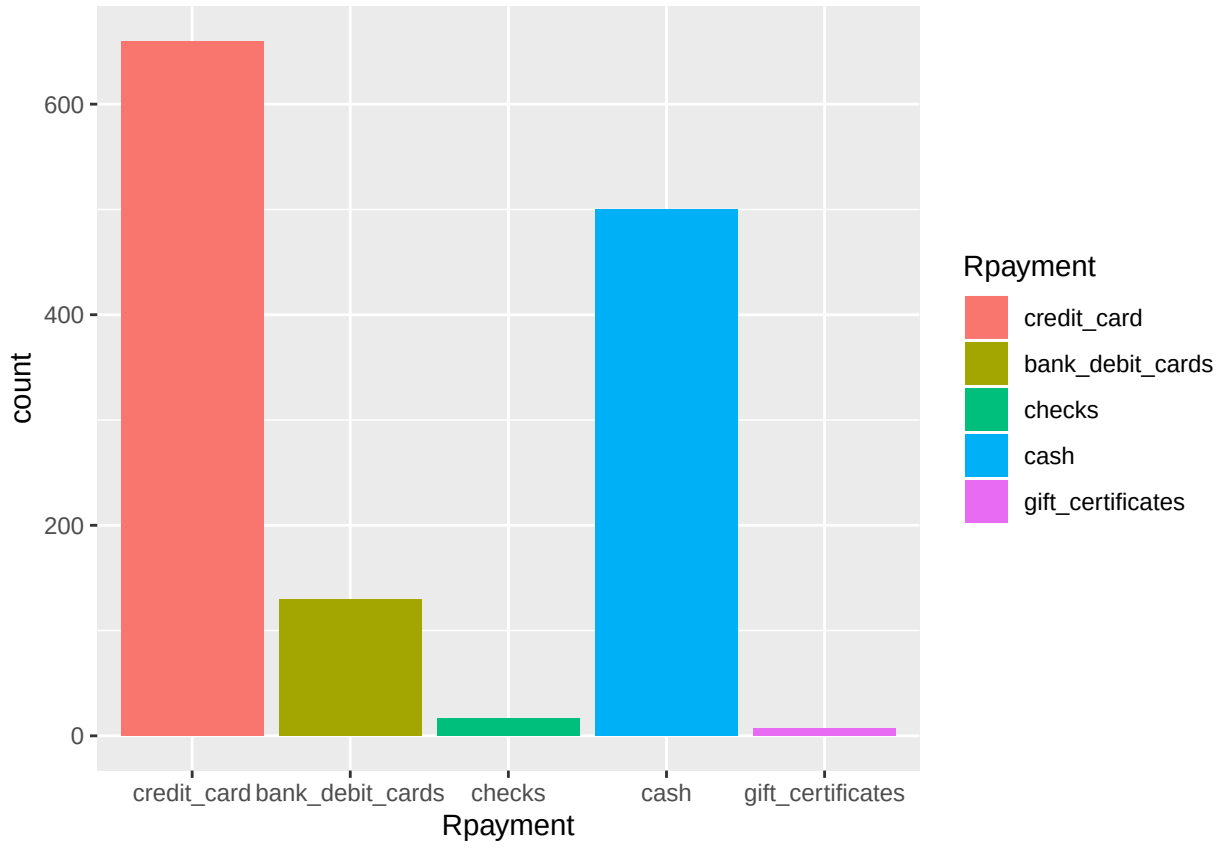
```
##  [1] "American_Express"    "bank_debit_cards"    "Carte_Blanche"
##  [4] "cash"                "checks"              "Diners_Club"
##  [7] "Discover"            "gift_certificates"   "Japan_Credit_Bureau"
## [10] "MasterCard-Eurocard" "Visa"                "VISA"
```

```
ggplot(accept, aes(x = Rpayment, fill=Rpayment)) + geom_bar()
```



Aufgrund der Anzahl der verschiedenen Kreditkarten ist die Anzahl der Zahlungsmethoden mit 12 Ausprägungen zu hoch. Aus diesem Grund werden die verschiedenen Kreditkarten zusammengefasst. Einfachheitshalber werden zusätzlich noch Carte_blanche und checks zusammengefasst.

```
accept$Rpayment = revalue(accept$Rpayment, c("American_Express"="credit_card", "MasterCard-Eurocard"="c

accept$Rpayment = revalue(accept$Rpayment, c("Carte_Blanche"="checks", "checks"="checks"))

ggplot(accept, aes(x = Rpayment, fill=Rpayment)) + geom_bar()
```




```
head(cuisine)
```

```
##   placeID       Rcuisine
## 1  135110        Spanish
## 2  135109        Italian
## 3  135107 Latin_American
## 4  135106        Mexican
## 5  135105      Fast_Food
## 6  135104        Mexican
```

```
dim(cuisine)
```

```
## [1] 916   2
```

```
levels(cuisine$Rcuisine)
```

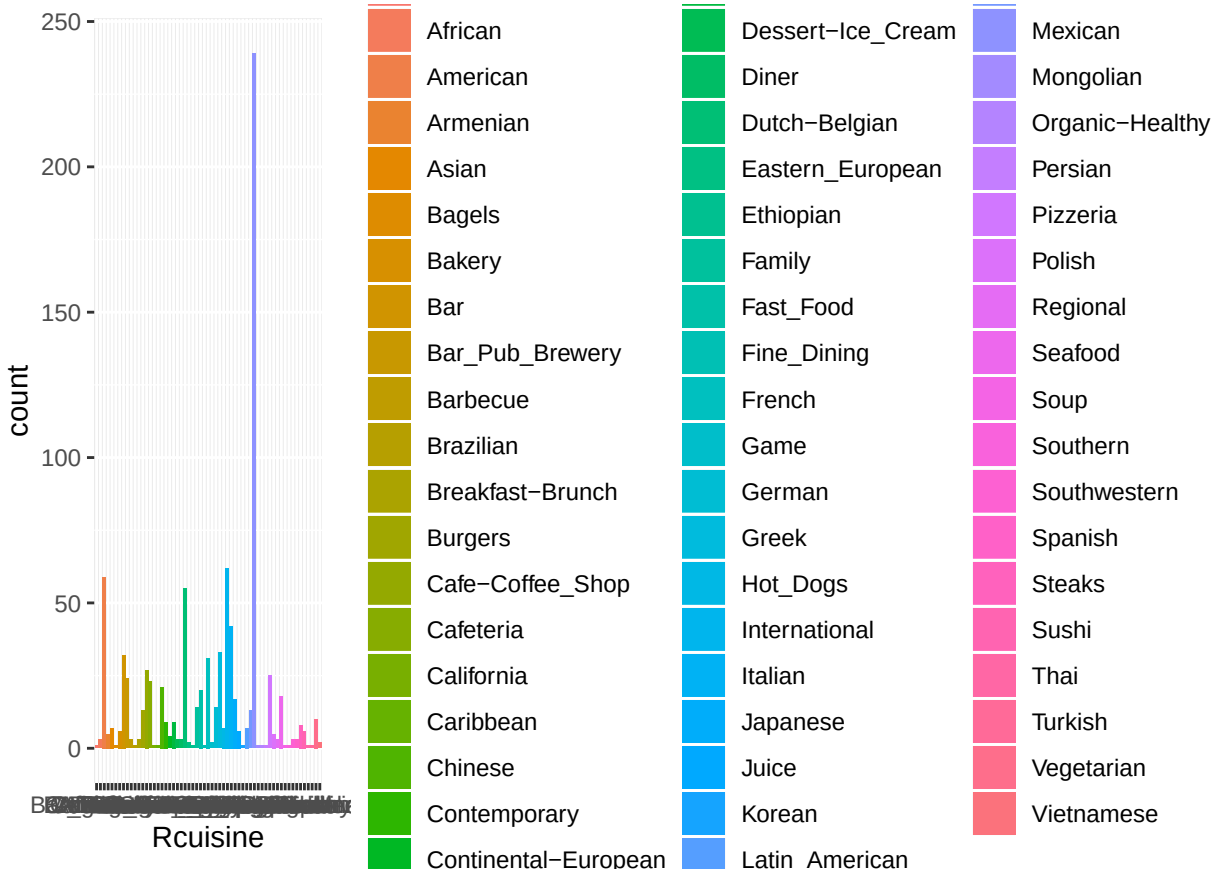
```
##  [1] "Afghan"               "African"              "American"
##  [4] "Armenian"             "Asian"                "Bagels"
##  [7] "Bakery"               "Bar"                  "Bar_Pub_Brewery"
## [10] "Barbecue"             "Brazilian"            "Breakfast-Brunch"
## [13] "Burgers"              "Cafe-Coffee_Shop"     "Cafeteria"
## [16] "California"           "Caribbean"            "Chinese"
## [19] "Contemporary"         "Continental-European" "Deli-Sandwiches"
## [22] "Dessert-Ice_Cream"    "Diner"                "Dutch-Belgian"
## [25] "Eastern_European"     "Ethiopian"            "Family"
```

```
## [28] "Fast_Food"            "Fine_Dining"          "French"
## [31] "Game"                 "German"               "Greek"
## [34] "Hot_Dogs"             "International"        "Italian"
## [37] "Japanese"             "Juice"                "Korean"
## [40] "Latin_American"       "Mediterranean"        "Mexican"
## [43] "Mongolian"            "Organic-Healthy"      "Persian"
## [46] "Pizzeria"             "Polish"               "Regional"
## [49] "Seafood"              "Soup"                 "Southern"
## [52] "Southwestern"         "Spanish"              "Steaks"
## [55] "Sushi"                "Thai"                 "Turkish"
## [58] "Vegetarian"           "Vietnamese"
```

```
summary(cuisine)
```

```
##     placeID                Rcuisine
##  Min.   :132001   Mexican      :239
##  1st Qu.:132323   International: 62
##  Median :132630   American     : 59
##  Mean   :132897   Dutch-Belgian: 55
##  3rd Qu.:132907   Italian      : 42
##  Max.   :135110   Greek        : 33
##                   (Other)      :426
```

```
ggplot(cuisine, aes(x = Rcuisine, fill=Rcuisine)) + geom_bar()
```



Die kategoriale Varible Rcuisine hat 59 Ausprägungen. Aus diesem Grund müssen diese zusammengefasst

werden. Zusammengefasst werden die Ausprägungen nach den Regionen. * Persian * Asia * American * European * South_American * African * International

```
cuisine$Rcuisine = revalue(cuisine$Rcuisine, c("Japanese"="Asian", "Chinese"="Asian",
                                               "Sushi"="Asian", "Korean"="Asian",
                                               "Mongolian"="Asian", "Thai"="Asian",
                                               "Asia"="Asian", "Vietnamese"="Asian",
                                               "Deli-Sandwiches"="Asian"))
```

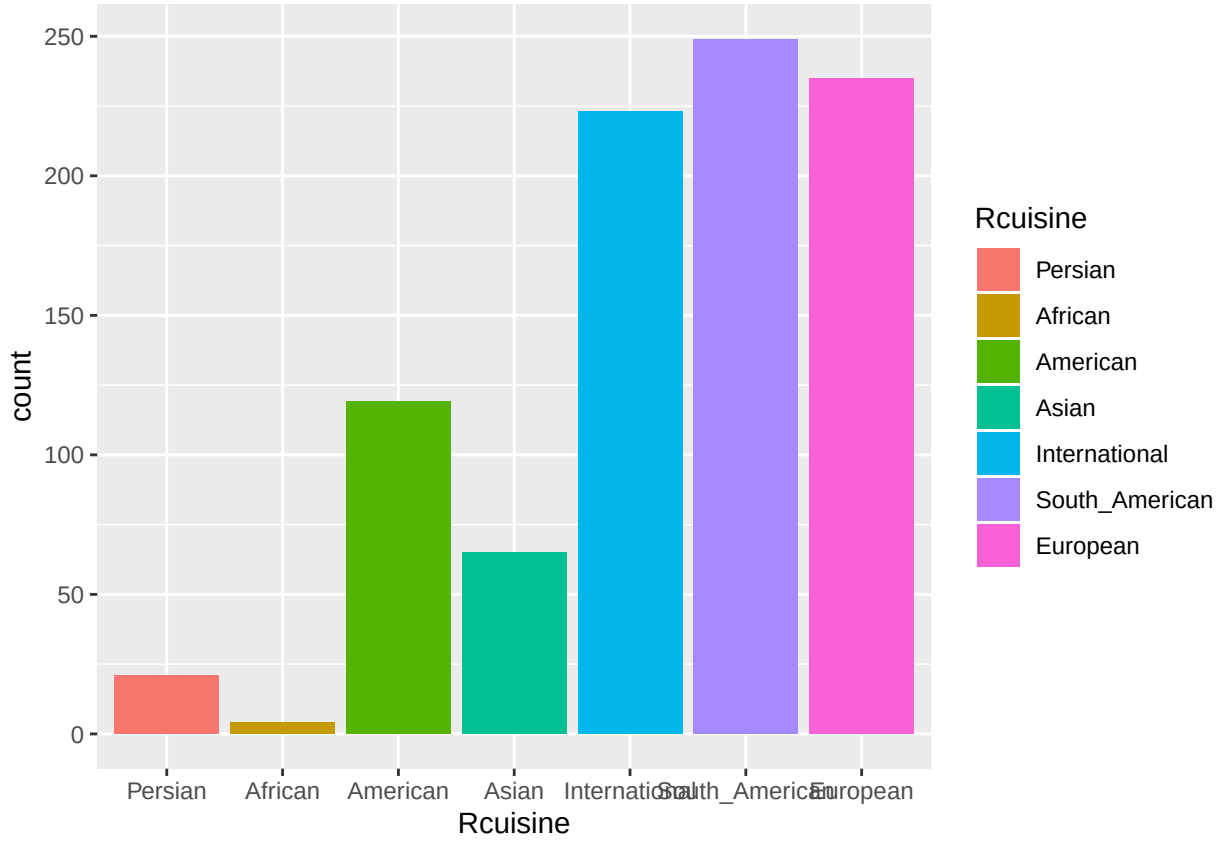
```
## The following `from` values were not present in `x`: Asia
```

```
cuisine$Rcuisine = revalue(cuisine$Rcuisine, c("Dutch-Belgian"="European",
                                               "Continental-European"="European",
                                               "Eastern_European"="European",
                                               "Greek"="European",
                                               "Spanish"="European", "French"="European",
                                               "German"="European", "Italian"="European",
                                               "Polish"="European", "Pizzeria"="European",
                                               "Dessert-Ice_Cream"="European",
                                               "Seafood"="European"))

cuisine$Rcuisine = revalue(cuisine$Rcuisine, c("Ethiopian"="African",
                                               "African"="African"))

cuisine$Rcuisine = revalue(cuisine$Rcuisine, c("Barbecue"="American",
                                               "Hot_Dogs"="American",
                                               "Steaks"="American",
                                               "American"="American",
                                               "Fast_Food"="American",
                                               "Burgers"="American",
                                               "California"="American",
                                               "Southwestern"="American",
                                               "Game"="American",
                                               "Diner"="American"))

cuisine$Rcuisine = revalue(cuisine$Rcuisine, c("Persian"="Persian",
                                               "Mediterranean"="Persian",
                                               "Turkish"="Persian",
                                               "Afghan"="Persian",
                                               "Armenian"="Persian"))

cuisine$Rcuisine = revalue(cuisine$Rcuisine, c("Brazilian"="South_American",
                                               "Caribbean"="South_American",
                                               "Southern"="South_American",
                                               "Mexican"="South_American",
                                               "Latin_American"="South_American"))

cuisine$Rcuisine = revalue(cuisine$Rcuisine, c("Bar"="International",
                                               "Contemporary"="International",
                                               "Fine_Dining"="International",
                                               "Vegetarian"="International",
                                               "Bakery"="International",
                                               "Cafe-Coffee_Shop"="International",
```

```
                                        "Organic-Healthy"="International",
                                        "Juice"="International",
                                        "Soup"="International",
                                        "Bagels"="International",
                                        "Bar_Pub_Brewery"="International",
                                        "Breakfast-Brunch"="International",
                                        "Cafeteria"="International",
                                        "Family"="International",
                                        "Regional"="International"))


levels(cuisine$Rcuisine)
```

```
## [1] "Persian"        "African"        "American"       "Asian"
## [5] "International"  "South_American" "European"
```

```
ggplot(cuisine, aes(x = Rcuisine, fill=Rcuisine)) + geom_bar()
```




```
head(hours)
```

```
##   placeID        hours                  days
## 1  135111 00:00-23:30; Mon;Tue;Wed;Thu;Fri;
## 2  135111 00:00-23:30;                 Sat;
## 3  135111 00:00-23:30;                 Sun;
## 4  135110 08:00-19:00; Mon;Tue;Wed;Thu;Fri;
```

```
## 5  135110 00:00-00:00;                Sat;
## 6  135110 00:00-00:00;                Sun;
```

```
dim(hours)
```

```
## [1] 2339    3
```

```
summary(hours)
```

```
##     placeID               hours                         days
##  Min.   :132012   00:00-23:30;: 681   Mon;Tue;Wed;Thu;Fri;:793
##  1st Qu.:132574   00:00-00:00;: 100   Sat;                 :783
##  Median :132785   17:00-22:00;:  56   Sun;                 :763
##  Mean   :133082   14:00-23:30;:  32
##  3rd Qu.:132984   09:00-23:30;:  31
##  Max.   :135111   11:00-21:00;:  31
##                   (Other)     :1408
```

```
#levels(hours$hours)
#ggplot(hours, aes(x = days, fill=hours)) + geom_bar()
```

Für die weitere Ausarbeitung werden die Öffnungszeiten nicht weiter berücksichtigt, da eine Gruppierung auf wenige Ausprägungen sich schwierig darstellt und somit für vorhersagen schwer zu verwenden ist.

```
head(parking)
```

```
##   placeID parking_lot
## 1  135111      public
## 2  135110        none
## 3  135109        none
## 4  135108        none
## 5  135107        none
## 6  135106        none
```

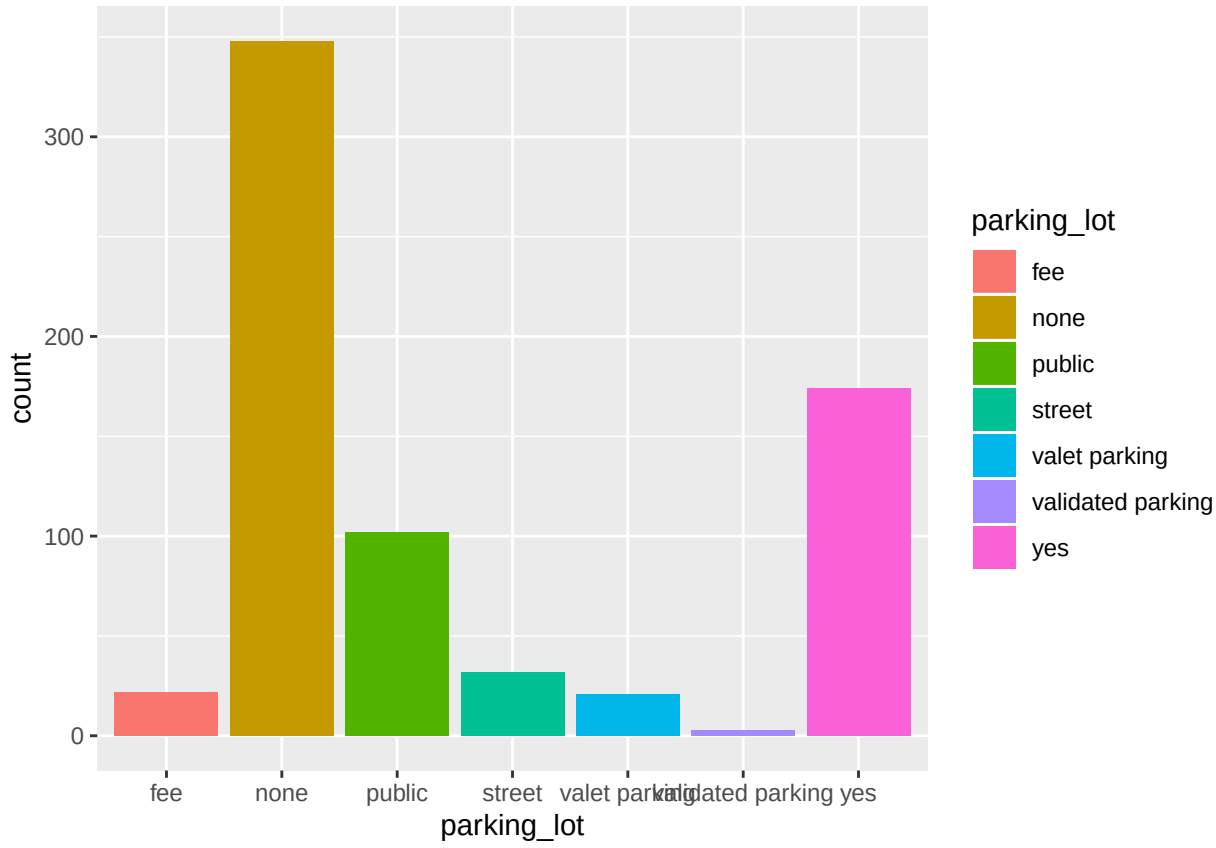
```
dim(parking)
```

```
## [1] 702   2
```

```
summary(parking)
```

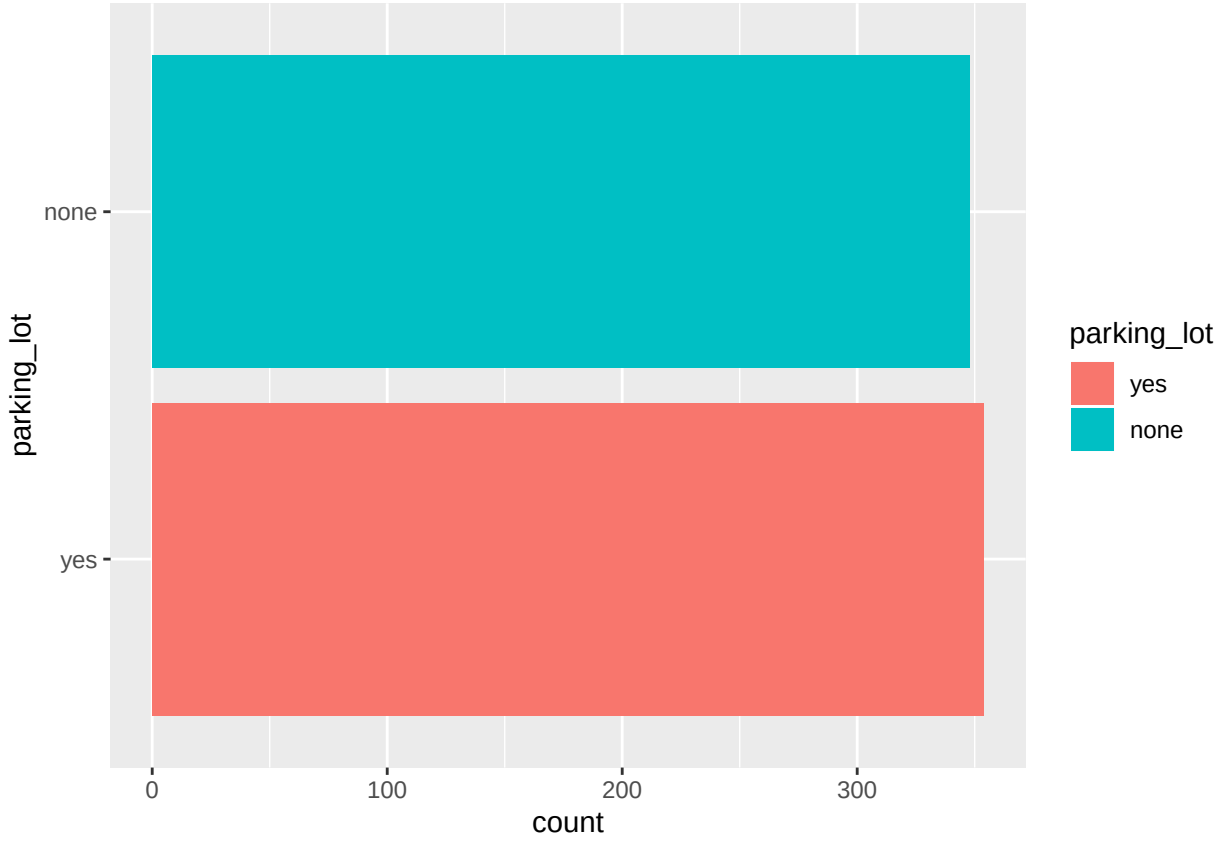
```
##     placeID                 parking_lot
##  Min.   :132012   fee              : 22
##  1st Qu.:132649   none             :348
##  Median :132826   public           :102
##  Mean   :133181   street           : 32
##  3rd Qu.:133009   valet parking    : 21
##  Max.   :135111   validated parking:  3
##                   yes              :174
```

```
ggplot(parking, aes(x = parking_lot, fill=parking_lot)) + geom_bar()
```



Die Parkplatzinformationen werden gruppiert nach ja und nein.

```
parking_grouped <- parking
parking_grouped$parking_lot = revalue(parking_grouped$parking_lot, c("fee"="yes", "public"="yes", "stree
ggplot(parking_grouped, aes(x = parking_lot, fill=parking_lot), ) + geom_bar() + coord_flip()
```




```
head(geoplaces)
```

```
##   placeID latitude  longitude
## 1  134999 18.91542  -99.18487
## 2  132825 22.14739 -100.98309
## 3  135106 22.14971 -100.97609
## 4  132667 23.75270  -99.16336
## 5  132613 23.75290  -99.16508
## 6  135040 22.13562 -100.96971
##                                      the_geom_meter
## 1 0101000020957F000088568DE356715AC138C0A525FC464A41
## 2 0101000020957F00001AD016568C4858C1243261274BA54B41
## 3 0101000020957F0000649D6F21634858C119AE9BF528A34B41
## 4 0101000020957F00005D67BCDDED8157C1222A2DC8D84D4941
## 5 0101000020957F00008EBA2D06DC8157C194E03B7B504E4941
## 6 0101000020957F00001B552189B84A58C15A2AAEFD2CA24B41
##                            name                                  address
## 1                Kiku Cuernavaca                                Revolucion
## 2                puesto de tacos esquina santos degollado y leon guzman
## 3     El Rinc n de San Francisco                          Universidad 169
## 4 little pizza Emilio Portes Gil                 calle emilio portes gil
## 5                  carnitas_mata                  lic. Emilio portes gil
## 6       Restaurant los Compadres           Camino a Simon Diaz 155 Centro
##              city           state country fax   zip           alcohol
## 1      Cuernavaca         Morelos  Mexico  NA  <NA> No_Alcohol_Served
## 2          s.l.p.          s.l.p.  mexico  NA 78280 No_Alcohol_Served
```

```
## 3 San Luis Potosi San Luis Potosi  Mexico  NA 78000         Wine-Beer
## 4       victoria       tamaulipas    <NA>  NA  <NA> No_Alcohol_Served
## 5        victoria      Tamaulipas  Mexico  NA  <NA> No_Alcohol_Served
## 6 San Luis Potosi             SLP  Mexico  NA 74000         Wine-Beer
##   smoking_area dress_code    accessibility  price                  url
## 1         none   informal no_accessibility medium kikucuernavaca.com.mx
## 2         none   informal       completely    low                 <NA>
## 3  only at bar   informal        partially medium                 <NA>
## 4         none   informal       completely    low                 <NA>
## 5    permitted   informal       completely medium                 <NA>
## 6         none   informal no_accessibility   high                 <NA>
##   Rambience franchise   area other_services
## 1  familiar         f closed           none
## 2  familiar         f   open           none
## 3  familiar         f   open           none
## 4  familiar         t closed           none
## 5  familiar         t closed           none
## 6  familiar         f closed           none
```

```
dim(geoplaces)
```

```
## [1] 130  21
```

```
summary(geoplaces)
```

```
##     placeID          latitude       longitude
##  Min.   :132560   Min.   :18.86   Min.   :-101.03
##  1st Qu.:132831   1st Qu.:22.14   1st Qu.:-100.99
##  Median :134994   Median :22.15   Median :-100.96
##  Mean   :134013   Mean   :21.86   Mean   :-100.34
##  3rd Qu.:135051   3rd Qu.:22.16   3rd Qu.: -99.22
##  Max.   :135109   Max.   :23.76   Max.   : -99.13
##
##                                          the_geom_meter
##  0101000020957F000000DD3546816E5AC119D4BD17FD544A41:  1
##  0101000020957F000003B195E25F8457C1C535BD04614B4941:  1
##  0101000020957F000004457BB7AA8657C15F10835CD9444941:  1
##  0101000020957F000005810F19B84858C136805B2745A74B41:  1
##  0101000020957F00000B6735CA004858C108FD525CB2A44B41:  1
##  0101000020957F00000F14BF6B2C8657C1963CCB8E5C464941:  1
##  (Other)                                          :124
##                         name                    address
##  Gorditas Dona Tota       :  2   Ricardo B. Anaya   : 3
##  Abondance Restaurante Bar:  1   Av. V. Carranza    : 2
##  Arrachela Grill          :  1   venustiano carranza: 2
##  Cabana Huasteca          :  1   16 de Septiembre   : 1
##  cafe ambar               :  1   1a. de Lozada 1    : 1
##  Cafe Chaires             :  1   (Other)            :94
##  (Other)                  :123   NA's               :27
##              city                state      country     fax
##  San Luis Potosi:64   SLP            :50   mexico:13   Mode:logical
##  Cuernavaca     :15   Morelos        :19   Mexico:89   NA's:130
##  victoria       :10   San Luis Potosi:14   NA's  :28
```

```
##  san luis potosi: 5   tamaulipas     : 9
##  Jiutepec       : 4   Tamaulipas     : 7
##  (Other)        :14   (Other)        :13
##  NA's           :18   NA's           :18
##       zip                  alcohol          smoking_area    dress_code
##  78000  :13   Full_Bar         : 9   none         :70    casual  : 10
##  78250  : 3   No_Alcohol_Served:87   not permitted:25    formal  :  2
##  78269  : 3   Wine-Beer        :34   only at bar  : 2    informal:118
##  62290  : 2                          permitted    : 9
##  78210  : 2                          section      :24
##  (Other):33
##  NA's   :74
##           accessibility    price                      url
##  completely      :45    high  :25   lacantinaslp.com     :  2
##  no_accessibility:76    low   :45   carlosandcharlies.com:  1
##  partially       : 9    medium:60   chilis.com.mx        :  1
##                                     eloceanodorado.com   :  1
##                                     kikucuernavaca.com.mx:  1
##                                     (Other)              :  8
##                                     NA's                 :116
##     Rambience   franchise     area      other_services
##  familiar:121   f:108     closed:115   Internet:  4
##  quiet   :  9   t: 22     open  : 15   none    :119
##                                       variety :  7
##
##
##
##
```

```
m <- leaflet() %>%
  addTiles() %>%  # Add default OpenStreetMap map tiles
  addMarkers(lng=geoplaces$longitude, lat=geoplaces$latitude, popup=geoplaces$name)
#m  # Print the map
```

1.4.2 Kunden-Daten

```
head(usercuisine)
```

```
##   userID         Rcuisine
## 1  U1001         American
## 2  U1002          Mexican
## 3  U1003          Mexican
## 4  U1004           Bakery
## 5  U1004 Breakfast-Brunch
## 6  U1004         Japanese
```

```
dim(usercuisine)
```

```
## [1] 330   2
```

```r
summary(usercuisine)
```

```
##      userID                Rcuisine
##  U1135  :103   Mexican          : 97
##  U1108  : 18   American         : 11
##  U1101  : 15   Cafeteria        :  9
##  U1016  : 14   Pizzeria         :  9
##  U1060  : 13   Cafe-Coffee_Shop:  8
##  U1008  : 10   Family           :  8
##  (Other):157   (Other)          :188
```

```r
levels(usercuisine$Rcuisine)
```

```
##   [1] "Afghan"               "African"              "American"
##   [4] "Armenian"             "Asian"                "Australian"
##   [7] "Austrian"             "Bagels"               "Bakery"
##  [10] "Bar"                  "Bar_Pub_Brewery"      "Barbecue"
##  [13] "Basque"               "Brazilian"            "Breakfast-Brunch"
##  [16] "British"              "Burgers"              "Burmese"
##  [19] "Cafe-Coffee_Shop"     "Cafeteria"            "Cajun-Creole"
##  [22] "California"           "Cambodian"            "Canadian"
##  [25] "Caribbean"            "Chilean"              "Chinese"
##  [28] "Contemporary"         "Continental-European" "Cuban"
##  [31] "Deli-Sandwiches"      "Dessert-Ice_Cream"    "Dim_Sum"
##  [34] "Diner"                "Doughnuts"            "Dutch-Belgian"
##  [37] "Eastern_European"     "Eclectic"             "Ethiopian"
##  [40] "Family"               "Fast_Food"            "Filipino"
##  [43] "Fine_Dining"          "French"               "Fusion"
##  [46] "Game"                 "German"               "Greek"
##  [49] "Hawaiian"             "Hot_Dogs"             "Hungarian"
##  [52] "Indian-Pakistani"     "Indigenous"           "Indonesian"
##  [55] "International"        "Irish"                "Israeli"
##  [58] "Italian"              "Jamaican"             "Japanese"
##  [61] "Juice"                "Korean"               "Kosher"
##  [64] "Latin_American"       "Lebanese"             "Malaysian"
##  [67] "Mediterranean"        "Mexican"              "Middle_Eastern"
##  [70] "Mongolian"            "Moroccan"             "North_African"
##  [73] "Organic-Healthy"      "Pacific_Northwest"    "Pacific_Rim"
##  [76] "Persian"              "Peruvian"             "Pizzeria"
##  [79] "Polish"               "Polynesian"           "Portuguese"
##  [82] "Regional"             "Romanian"             "Russian-Ukrainian"
##  [85] "Scandinavian"         "Seafood"              "Soup"
##  [88] "Southeast_Asian"      "Southern"             "Southwestern"
##  [91] "Spanish"              "Steaks"               "Sushi"
##  [94] "Swiss"                "Tapas"                "Tea_House"
##  [97] "Tex-Mex"              "Thai"                 "Tibetan"
## [100] "Tunisian"             "Turkish"              "Vegetarian"
## [103] "Vietnamese"
```

Um die Ausprägungen der Rcuisine zu reduzieren werden die gleichen Levels wie obrig verwendet.

```
usercuisine$Rcuisine = revalue(usercuisine$Rcuisine, c("Japanese"="Asian", "Chinese"="Asian",
                                                       "Sushi"="Asian", "Korean"="Asian",
                                                       "Mongolian"="Asian", "Thai"="Asian",
                                                       "Asia"="Asian", "Vietnamese"="Asian",
                                                       "Deli-Sandwiches"="Asian",
                                                       "Southeast_Asian"="Asian",
                                                       "Burmese"="Asian", "Cambodian"="Asian",
                                                       "Malaysian"="Asian", "Dim_Sum"="Asian", "Indonesian"="As
```

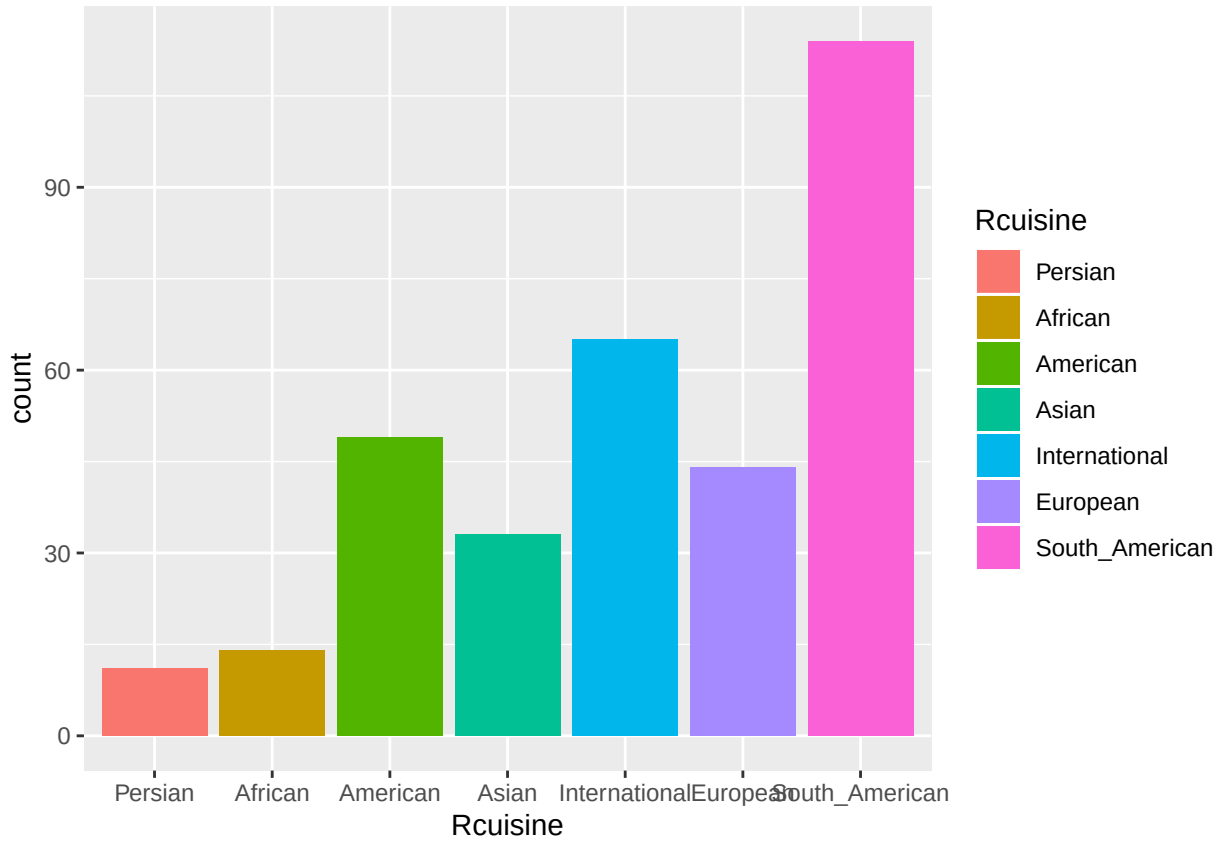
```
## The following `from` values were not present in `x`: Asia
```

```
usercuisine$Rcuisine = revalue(usercuisine$Rcuisine, c("Dutch-Belgian"="European",
                                                       "Continental-European"="European",
                                                       "Eastern_European"="European",
                                                       "Greek"="European",
                                                       "Spanish"="European", "French"="European",
                                                       "German"="European", "Italian"="European",
                                                       "Polish"="European", "Pizzeria"="European",
                                                       "Dessert-Ice_Cream"="European",
                                                       "Seafood"="European",
                                                       "British"="European",
                                                       "Irish"="European",
                                                       "Swiss"="European",
                                                       "Filipino"="European", "Austrian"="European", "Hungarian
                                                       "Portuguese"="European", "Romanian"="European", "Basque"=
                                                       "Scandinavian"="European", "Russian-Ukrainian"="European

usercuisine$Rcuisine = revalue(usercuisine$Rcuisine, c("Ethiopian"="African",
                                                       "African"="African",
                                                       "North_African"="African",
                                                       "Israeli"="African",
                                                       "Jamaican"="African", "Lebanese"="African", "Tibetan"="A
                                                       "Middle_Eastern"="African", "Moroccan"="African"))

usercuisine$Rcuisine = revalue(usercuisine$Rcuisine, c("Barbecue"="American",
                                                       "Hot_Dogs"="American",
                                                       "Steaks"="American",
                                                       "American"="American",
                                                       "Fast_Food"="American",
                                                       "Burgers"="American",
                                                       "California"="American",
                                                       "Southwestern"="American",
                                                       "Game"="American",
                                                       "Diner"="American",
                                                       "Doughnuts"="American",
                                                       "Pacific_Northwest"="American",
                                                       "Cajun-Creole"="American",
                                                       "Pacific_Rim"="American",
                                                       "Canadian"="American",
                                                       "Hawaiian"="American", "Indigenous"="American"))

usercuisine$Rcuisine = revalue(usercuisine$Rcuisine, c("Persian"="Persian",
                                                       "Mediterranean"="Persian",
```

```
                                                "Turkish"="Persian",
                                                "Afghan"="Persian",
                                                "Armenian"="Persian",
                                                "Indian-Pakistani"="Persian"))

usercuisine$Rcuisine = revalue(usercuisine$Rcuisine, c("Brazilian"="South_American",
                                                "Caribbean"="South_American",
                                                "Southern"="South_American",
                                                "Mexican"="South_American",
                                                "Latin_American"="South_American", "Peruvian"="South_Amer
                                                "Tapas"="South_American", "Tex-Mex"="South_American",
                                                "Chilean"="South_American", "Cuban"="South_American"))

usercuisine$Rcuisine = revalue(usercuisine$Rcuisine, c("Bar"="International",
                                                "Contemporary"="International",
                                                "Fine_Dining"="International",
                                                "Vegetarian"="International",
                                                "Bakery"="International",
                                                "Cafe-Coffee_Shop"="International",
                                                "Organic-Healthy"="International",
                                                "Juice"="International",
                                                "Soup"="International",
                                                "Bagels"="International",
                                                "Bar_Pub_Brewery"="International",
                                                "Breakfast-Brunch"="International",
                                                "Cafeteria"="International",
                                                "Family"="International",
                                                "Regional"="International",
                                                "Eclectic"="International", "Fusion"="International",
                                                "Tea_House"="International",
                                                "Australian"="International",
                                                "Kosher"="International", "Polynesian"="International"))


levels(usercuisine$Rcuisine)
```

```
## [1] "Persian"        "African"        "American"       "Asian"
## [5] "International"  "European"       "South_American"
```

```
ggplot(usercuisine, aes(x = Rcuisine, fill=Rcuisine)) + geom_bar()
```




```
head(userpayment)
```

```
##   userID         Upayment
## 1  U1001             cash
## 2  U1002             cash
## 3  U1003             cash
## 4  U1004             cash
## 5  U1004 bank_debit_cards
## 6  U1005             cash
```

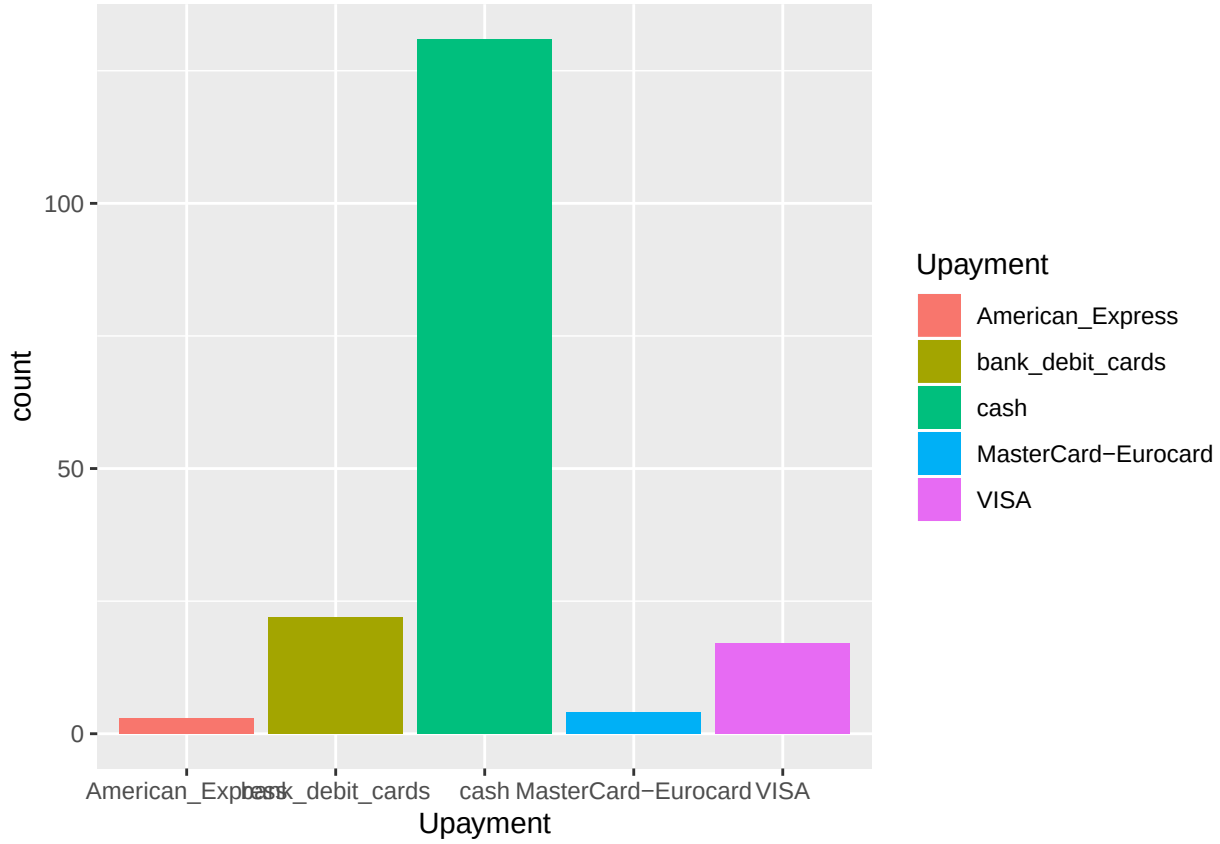
```
dim(userpayment)
```

```
## [1] 177   2
```

```
summary(userpayment)
```

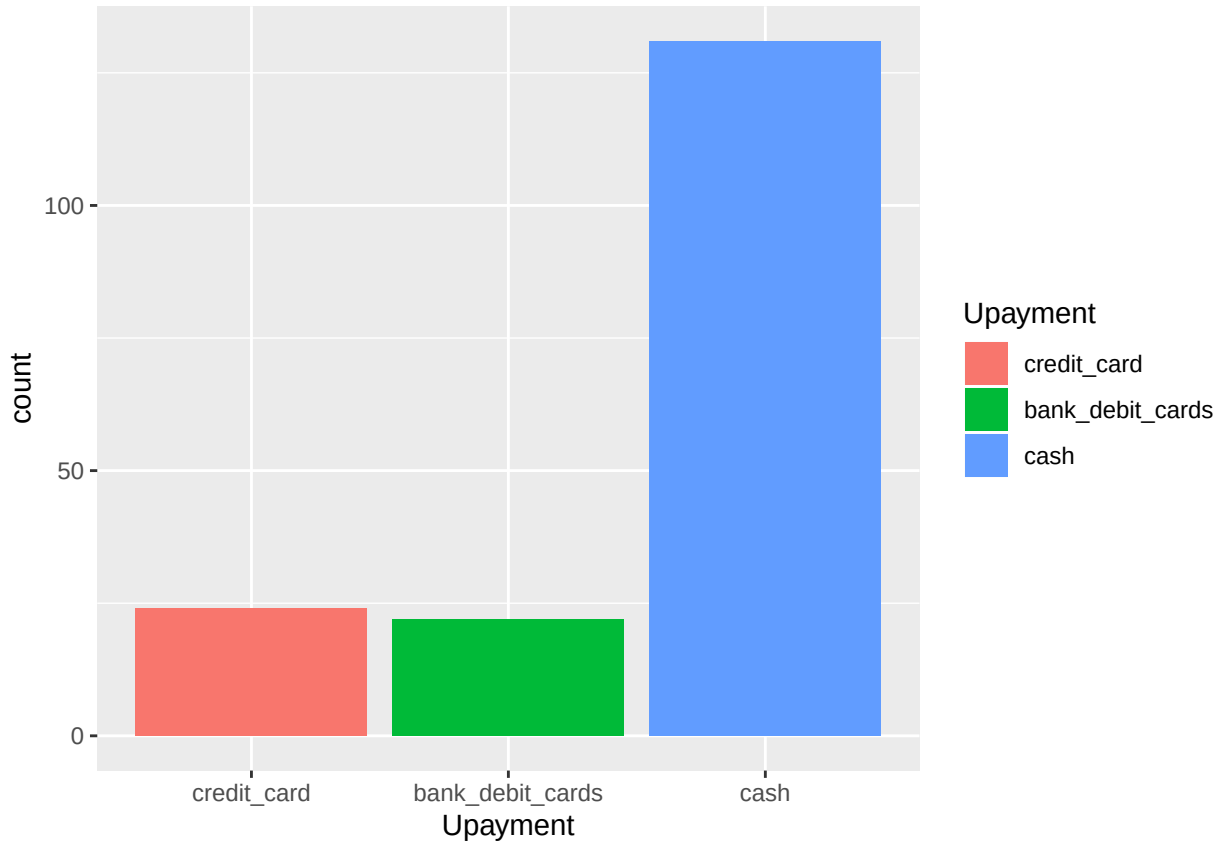
```
##      userID                   Upayment
##  U1041  :  4   American_Express   :  3
##  U1044  :  4   bank_debit_cards   : 22
##  U1076  :  3   cash               :131
##  U1077  :  3   MasterCard-Eurocard:  4
##  U1078  :  3   VISA               : 17
##  U1086  :  3
##  (Other):157
```

```
ggplot(userpayment, aes(x = Upayment, fill=Upayment)) + geom_bar()
```



Auch bei den Zahlungsmethoden werden die Kreditkarten wie obrig zusammengefasst.

```
userpayment$Upayment = revalue(userpayment$Upayment, c("American_Express"="credit_card", "MasterCard-Eu
ggplot(userpayment, aes(x = Upayment, fill=Upayment)) + geom_bar()
```




```
head(userprofile)
```

```
##   userID latitude longitude smoker    drink_level dress_preference
## 1  U1001 22.14000 -100.9788  false     abstemious         informal
## 2  U1002 22.15009 -100.9833  false     abstemious         informal
## 3  U1003 22.11985 -100.9465  false social drinker           formal
## 4  U1004 18.86700  -99.1830  false     abstemious         informal
## 5  U1005 22.18348 -100.9599  false     abstemious    no preference
## 6  U1006 22.15000 -100.9830   true social drinker    no preference
##   ambience transport marital_status       hijos birth_year   interest
## 1   family   on foot         single independent       1989    variety
## 2   family    public         single independent       1990 technology
## 3   family    public         single independent       1989       none
## 4   family    public         single independent       1940    variety
## 5   family    public         single independent       1992       none
## 6  friends car owner         single independent       1989    variety
##           personality religion     activity color weight budget height
## 1   thrifty-protector     none      student black     69 medium   1.77
## 2 hunter-ostentatious Catholic      student   red     40    low   1.87
## 3         hard-worker Catholic      student  blue     60    low   1.69
## 4         hard-worker     none professional green     44 medium   1.53
## 5   thrifty-protector Catholic      student black     65 medium   1.69
## 6         hard-worker     none      student  blue     75 medium   1.80
```

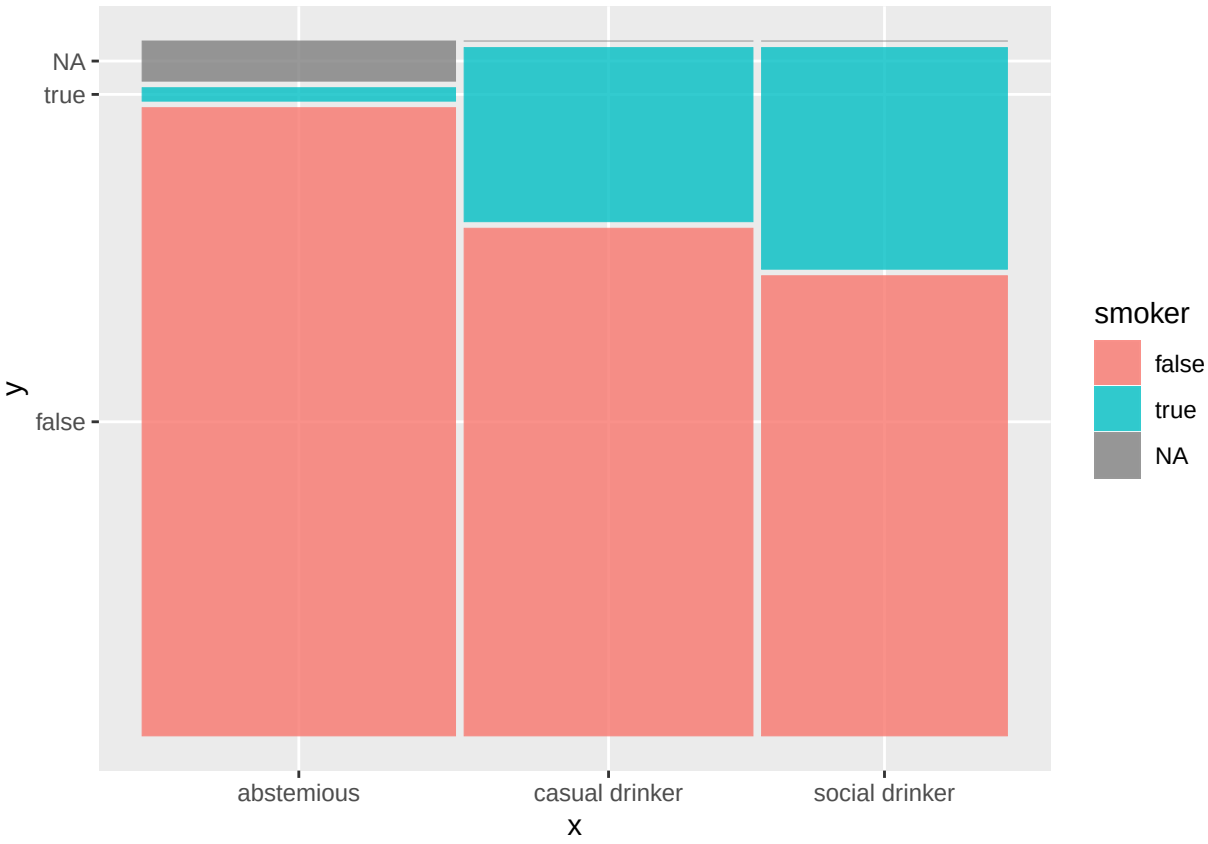
```r
dim(userprofile)
```

```
## [1] 138  19
```

```r
summary(userprofile)
```

```
##      userID       latitude       longitude         smoker
##  U1001  :  1   Min.   :18.81   Min.   :-101.05   false:109
##  U1002  :  1   1st Qu.:22.13   1st Qu.:-100.98   true : 26
##  U1003  :  1   Median :22.15   Median :-100.94   NA's :  3
##  U1004  :  1   Mean   :21.81   Mean   :-100.29
##  U1005  :  1   3rd Qu.:22.19   3rd Qu.: -99.18
##  U1006  :  1   Max.   :23.77   Max.   : -99.07
##  (Other):132
##          drink_level      dress_preference     ambience      transport
##  abstemious    :51   elegant      : 4      family  :70   car owner:35
##  casual drinker:47   formal       :41      friends :46   on foot  :14
##  social drinker:40   informal     :35      solitary:16   public   :82
##                      no preference:53      NA's    : 6   NA's     : 7
##                      NA's         : 5
##
##
##  marital_status         hijos       birth_year          interest
##  married: 10    dependent  :  3   Min.   :1930   eco-friendly:16
##  single :122    independent:113   1st Qu.:1987   none        :30
##  widow  :  2    kids       : 11   Median :1989   retro       : 6
##  NA's   :  4    NA's       : 11   Mean   :1985   technology  :36
##                                   3rd Qu.:1991   variety     :50
##                                   Max.   :1994
##
##               personality      religion           activity       color
##  conformist         : 7   Catholic :99   professional : 15   blue   :45
##  hard-worker        :61   Christian: 7   student      :113   black  :21
##  hunter-ostentatious:12   Jewish   : 1   unemployed   :  2   green  :19
##  thrifty-protector  :58   Mormon   : 1   working-class:  1   red    :15
##                           none     :30   NA's         :  7   yellow :12
##                                                             purple :11
##                                                             (Other):15
##      weight          budget       height
##  Min.   : 40.00   high  : 5   Min.   :1.200
##  1st Qu.: 53.00   low   :35   1st Qu.:1.600
##  Median : 65.00   medium:91   Median :1.690
##  Mean   : 64.87   NA's  : 7   Mean   :1.668
##  3rd Qu.: 74.75               3rd Qu.:1.750
##  Max.   :120.00               Max.   :2.000
##
```

```r
ggplot(userprofile) + geom_mosaic(aes(product(smoker,drink_level), fill = smoker))
```



1.4.3 Rating

```
head(rating)
```

```
##   userID placeID rating food_rating service_rating
## 1  U1077  135085      2           2              2
## 2  U1077  135038      2           2              1
## 3  U1077  132825      2           2              2
## 4  U1077  135060      1           2              2
## 5  U1068  135104      1           1              2
## 6  U1068  132740      0           0              0
```

```
dim(rating)
```

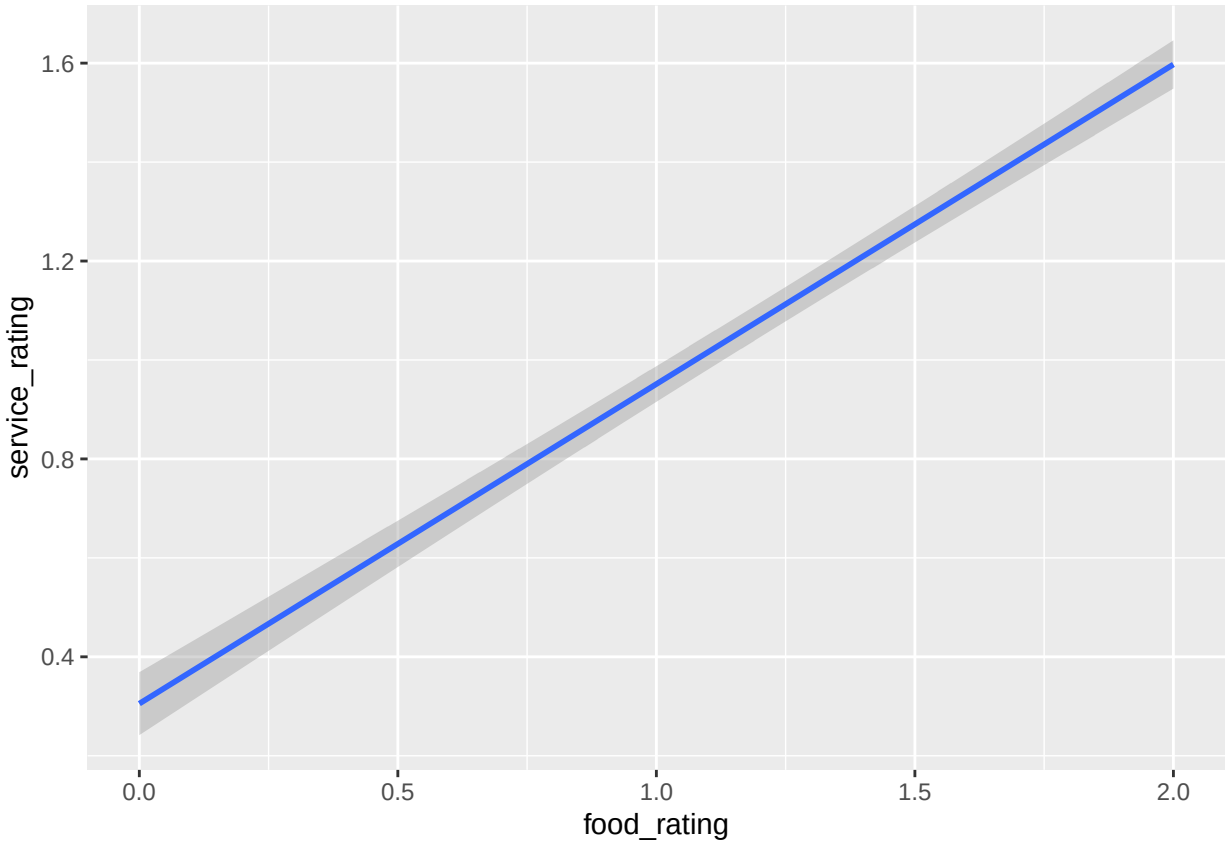
```
## [1] 1161    5
```

```
summary(rating)
```

```
##      userID        placeID          rating     food_rating
##  U1061  :  18   Min.   :132560   Min.   :0.0   Min.   :0.000
##  U1106  :  18   1st Qu.:132856   1st Qu.:1.0   1st Qu.:1.000
##  U1134  :  16   Median :135030   Median :1.0   Median :1.000
##  U1024  :  15   Mean   :134192   Mean   :1.2   Mean   :1.215
```

```
##  U1022  :  14   3rd Qu.:135059   3rd Qu.:2.0   3rd Qu.:2.000
##  U1089  :  14   Max.   :135109   Max.   :2.0   Max.   :2.000
##  (Other):1066
##  service_rating
##  Min.   :0.00
##  1st Qu.:0.00
##  Median :1.00
##  Mean   :1.09
##  3rd Qu.:2.00
##  Max.   :2.00
##
```

```
ggplot(rating, aes(food_rating, service_rating)) +
  geom_smooth(method = "lm")
```



1.5 Explorative Datenanalyse, speziell Visualisierung [20%]

1.5.1 Restaurant Data

Eigenheiten der Restaurants

```
cuisine_detail <- cuisine %>%
  join(geoplaces)
```

```
## Joining by: placeID
```

```
cuisine_detail <- cuisine_detail %>%
  filter(!is.na(name)) %>%
  select(placeID, name, Rcuisine, alcohol, smoking_area, dress_code, accessibility, price, Rambience)

head(cuisine_detail)
```

```
##   placeID                        name       Rcuisine           alcohol
## 1  135109                    Paniroles       European         Wine-Beer
## 2  135106  El Rinc n de San Francisco South_American         Wine-Beer
## 3  135104                        vips South_American          Full_Bar
## 4  135088           Cafeteria cenidet  International No_Alcohol_Served
## 5  135086 Mcdonalds Parque Tangamanga       American No_Alcohol_Served
## 6  135086 Mcdonalds Parque Tangamanga       American No_Alcohol_Served
##    smoking_area dress_code    accessibility  price Rambience
## 1 not permitted   informal no_accessibility medium     quiet
## 2   only at bar   informal        partially medium  familiar
## 3 not permitted   informal       completely medium  familiar
## 4 not permitted   informal no_accessibility    low     quiet
## 5 not permitted   informal no_accessibility medium  familiar
## 6 not permitted   informal no_accessibility medium  familiar
```

Überblick über einzelne Ausprägungen in den Bereichen Alcohol, Smoking und Price

```
# Generating Distribution Tables
## Alcohol
cuisine_detail_dist_alc<- cuisine_detail %>%
  distinct(Rcuisine, alcohol)

cuisine_detail_dist_alc <- gather(cuisine_detail_dist_alc, key, value, -Rcuisine) %>%
  count(Rcuisine, value) %>%
  spread(value, n, fill = 0) %>%
  group_by(Rcuisine) %>%
  rename(Alc_Full_Bar = Full_Bar, Alc_No_Alcohol_Served = No_Alcohol_Served, Alc_Wine_Beer = "Wine-Beer

head(cuisine_detail_dist_alc)
```

```
## # A tibble: 6 x 4
## # Groups:   Rcuisine [6]
##   Rcuisine       Alc_Full_Bar Alc_No_Alcohol_Served Alc_Wine_Beer
##   <fct>                 <dbl>                 <dbl>         <dbl>
## 1 Persian                   0                     1             0
## 2 American                  0                     1             0
## 3 Asian                     0                     1             1
## 4 International             1                     1             1
## 5 South_American            1                     1             1
## 6 European                  0                     1             1
```

```
## Smoking
cuisine_detail_dist_smoking<- cuisine_detail %>%
  distinct(Rcuisine, smoking_area)

cuisine_detail_dist_smoking <- gather(cuisine_detail_dist_smoking, key, value, -Rcuisine) %>%
```

```
  count(Rcuisine, value) %>%
  spread(value, n, fill = 0) %>%
  group_by(Rcuisine) %>%
  rename(smoking_not_permitted = "not permitted", smoking_only_at_bar = "only at bar", smoking_none = no

head(cuisine_detail_dist_smoking)
```

```
## # A tibble: 6 x 6
## # Groups:   Rcuisine [6]
##   Rcuisine smoking_none smoking_not_per~ smoking_only_at~ smoking_permitt~
##   <fct>           <dbl>            <dbl>            <dbl>            <dbl>
## 1 Persian             1                0                0                0
## 2 American            1                1                0                0
## 3 Asian               1                0                0                1
## 4 Interna~            1                1                0                1
## 5 South_A~            1                1                1                1
## 6 European            1                1                0                0
## # ... with 1 more variable: smoking_section <dbl>
```

```
## Price
cuisine_detail_dist_price<- cuisine_detail %>%
  distinct(Rcuisine, price)

cuisine_detail_dist_price <- gather(cuisine_detail_dist_price, key, value, -Rcuisine) %>%
  count(Rcuisine, value) %>%
  spread(value, n, fill = 0) %>%
  group_by(Rcuisine) %>%
  rename(price_low = low, price_medium = medium, price_high = high)

head(cuisine_detail_dist_price)
```

```
## # A tibble: 6 x 4
## # Groups:   Rcuisine [6]
##   Rcuisine       price_high price_low price_medium
##   <fct>               <dbl>     <dbl>        <dbl>
## 1 Persian                 0         1            0
## 2 American                1         1            1
## 3 Asian                   1         0            1
## 4 International           1         1            1
## 5 South_American          1         1            1
## 6 European                1         1            1
```

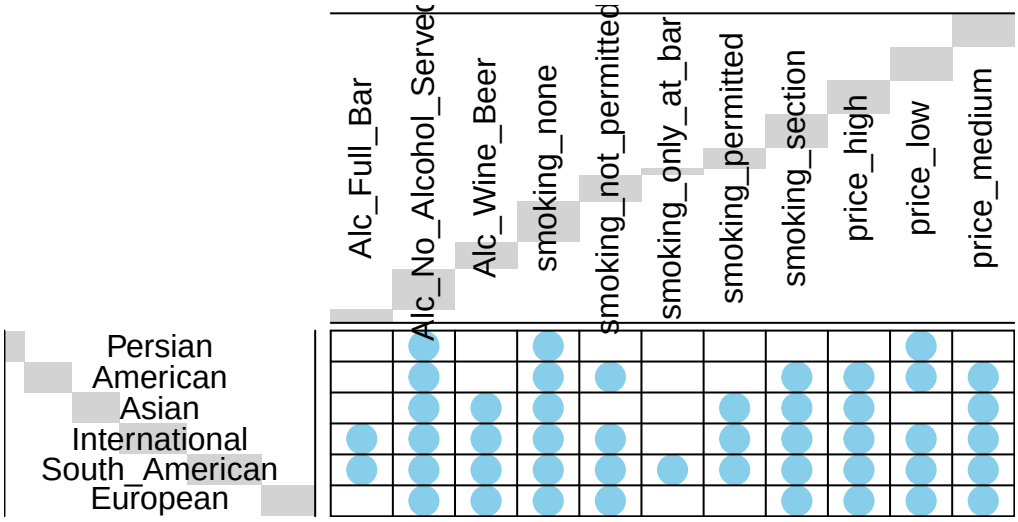
```
# JOINING TABLES

dt_dist <- cbind(cuisine_detail_dist_alc, cuisine_detail_dist_smoking, cuisine_detail_dist_price)
dt_dist$Rcuisine1 <- NULL
dt_dist$Rcuisine2 <- NULL
dt <- column_to_rownames(dt_dist, 'Rcuisine')
dt <- as.table(as.matrix(dt))
head(dt)
```

```
##                Alc_Full_Bar Alc_No_Alcohol_Served Alc_Wine_Beer
```

```
## Persian                   0                    1             0
## American                  0                    1             0
## Asian                     0                    1             1
## International             1                    1             1
## South_American            1                    1             1
## European                  0                    1             1
##                 smoking_none smoking_not_permitted smoking_only_at_bar
## Persian                    1                    0                   0
## American                   1                    1                   0
## Asian                      1                    0                   0
## International              1                    1                   0
## South_American             1                    1                   1
## European                   1                    1                   0
##                 smoking_permitted smoking_section price_high price_low
## Persian                         0               0          0         1
## American                        0               1          1         1
## Asian                           1               1          1         0
## International                   1               1          1         1
## South_American                  1               1          1         1
## European                        0               1          1         1
##                 price_medium
## Persian                    0
## American                   1
## Asian                      1
## International              1
## South_American             1
## European                   1
```

```
balloonplot(t(dt), main ="Distribution Smoking, Alcohol, Pricing Grouped By the Cuisines", xlab ="", yla
            label = FALSE, show.margins = FALSE, colsrt=90, rowmar=5, colmar=10)
```

Distribution Smoking, Alcohol, Pricing Grouped By the Cuisines



Verteilung der Preisklassen in den jeweiligen Cuisines

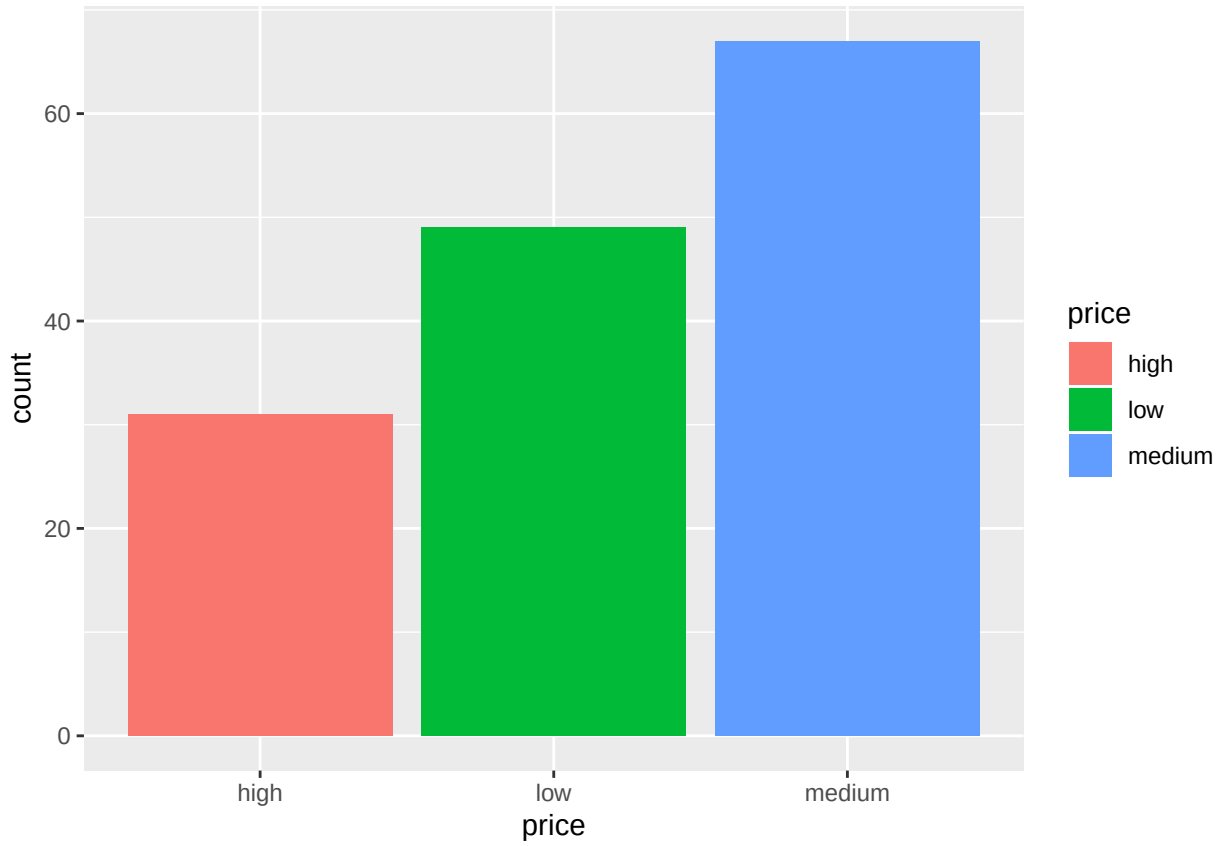
```
detailed_price <- geoplaces %>%
  join(cuisine)
```

```
## Joining by: placeID
```

```
head(detailed_price)
```

```
##   placeID latitude  longitude
## 1  134999 18.91542  -99.18487
## 2  132825 22.14739 -100.98309
## 3  135106 22.14971 -100.97609
## 4  132667 23.75270  -99.16336
## 5  132613 23.75290  -99.16508
## 6  135040 22.13562 -100.96971
##                                       the_geom_meter
## 1 0101000020957F000088568DE356715AC138C0A525FC464A41
## 2 0101000020957F00001AD016568C4858C1243261274BA54B41
## 3 0101000020957F0000649D6F21634858C119AE9BF528A34B41
## 4 0101000020957F00005D67BCDDED8157C1222A2DC8D84D4941
## 5 0101000020957F00008EBA2D06DC8157C194E03B7B504E4941
## 6 0101000020957F00001B552189B84A58C15A2AAEFD2CA24B41
##                             name                                  address
## 1                Kiku Cuernavaca                                Revolucion
## 2                puesto de tacos esquina santos degollado y leon guzman
## 3     El Rinc n de San Francisco                           Universidad 169
## 4 little pizza Emilio Portes Gil                  calle emilio portes gil
## 5                  carnitas_mata                   lic. Emilio portes gil
## 6       Restaurant los Compadres           Camino a Simon Diaz 155 Centro
##              city           state country fax   zip           alcohol
## 1      Cuernavaca         Morelos  Mexico  NA  <NA> No_Alcohol_Served
## 2          s.l.p.          s.l.p.  mexico  NA 78280 No_Alcohol_Served
## 3 San Luis Potosi San Luis Potosi  Mexico  NA 78000         Wine-Beer
## 4       victoria       tamaulipas    <NA>  NA  <NA> No_Alcohol_Served
## 5        victoria      Tamaulipas  Mexico  NA  <NA> No_Alcohol_Served
## 6 San Luis Potosi             SLP  Mexico  NA 74000         Wine-Beer
##   smoking_area dress_code    accessibility  price                   url
## 1         none   informal no_accessibility medium kikucuernavaca.com.mx
## 2         none   informal       completely    low                  <NA>
## 3  only at bar   informal        partially medium                  <NA>
## 4         none   informal       completely    low                  <NA>
## 5    permitted   informal       completely medium                  <NA>
## 6         none   informal no_accessibility   high                  <NA>
##   Rambience franchise   area other_services       Rcuisine
## 1  familiar         f closed           none          Asian
## 2  familiar         f   open           none South_American
## 3  familiar         f   open           none South_American
## 4  familiar         t closed           none        Persian
## 5  familiar         t closed           none South_American
## 6  familiar         f closed           none           <NA>
```

```
ggplot(detailed_price, aes(x = price, fill=price)) + geom_bar()
```




```
ggplot(detailed_price, aes(x = price, fill=price)) + geom_bar() + facet_wrap(.~Rcuisine)
```

1.5.2 Kunden Data

Bewertugen der Places mit cuisine und name

```
user_detail <- userprofile %>%
  join(usercuisine) %>%
  join(userpayment)
```

```
## Joining by: userID
## Joining by: userID
```

```
head(user_detail)
```

```
##   userID latitude longitude smoker    drink_level dress_preference
## 1  U1001 22.14000 -100.9788  false     abstemious         informal
## 2  U1002 22.15009 -100.9833  false     abstemious         informal
## 3  U1003 22.11985 -100.9465  false social drinker           formal
## 4  U1004 18.86700  -99.1830  false     abstemious         informal
## 5  U1004 18.86700  -99.1830  false     abstemious         informal
## 6  U1004 18.86700  -99.1830  false     abstemious         informal
##   ambience transport marital_status       hijos birth_year   interest
## 1   family   on foot         single independent       1989    variety
## 2   family    public         single independent       1990 technology
## 3   family    public         single independent       1989       none
## 4   family    public         single independent       1940    variety
```

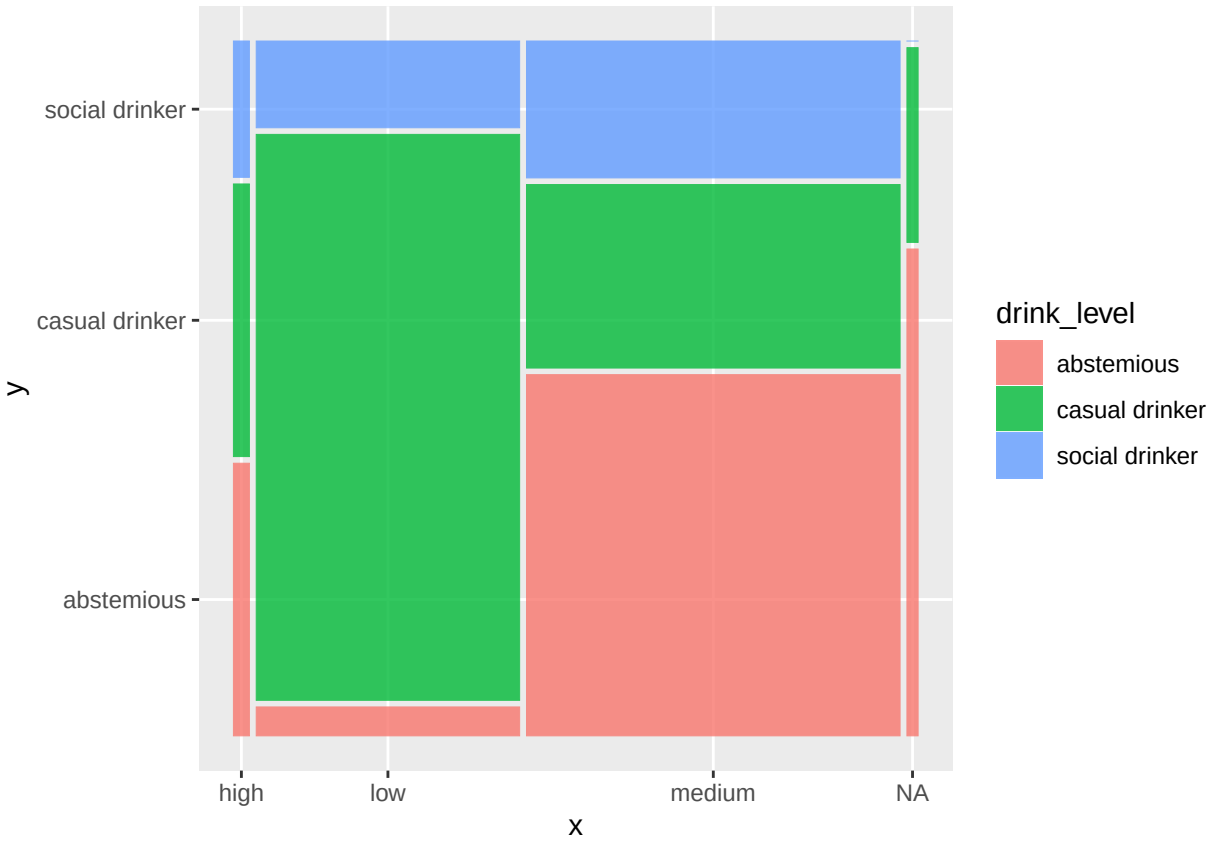
```
## 5   family    public         single independent       1940    variety
## 6   family    public         single independent       1940    variety
##           personality religion     activity color weight budget height
## 1   thrifty-protector     none      student black     69 medium   1.77
## 2 hunter-ostentatious Catholic      student   red     40    low   1.87
## 3         hard-worker Catholic      student  blue     60    low   1.69
## 4         hard-worker     none professional green     44 medium   1.53
## 5         hard-worker     none professional green     44 medium   1.53
## 6         hard-worker     none professional green     44 medium   1.53
##         Rcuisine         Upayment
## 1       American             cash
## 2 South_American             cash
## 3 South_American             cash
## 4  International             cash
## 5  International bank_debit_cards
## 6  International             cash
```

Standort der Kunden und der Restaurants

```
customers = makeIcon("user_icon.png", 50, 50)
restaurants = makeIcon("restaurant_icon.png", 50, 50)
m <- leaflet() %>%
  addTiles() %>%  # Add default OpenStreetMap map tiles
  addMarkers(lng=userprofile$longitude, lat=userprofile$latitude, popup=userprofile$userID, icon = custo
  addMarkers(lng = geoplaces$longitude, lat = geoplaces$latitude, icon = restaurants, popup = geoplaces$
#m  # Print the map
```

Einfluss von dem Trinkverhalten auf das Budget.

```
ggplot(user_detail) + geom_mosaic(aes(product(drink_level,budget), fill = drink_level))
```

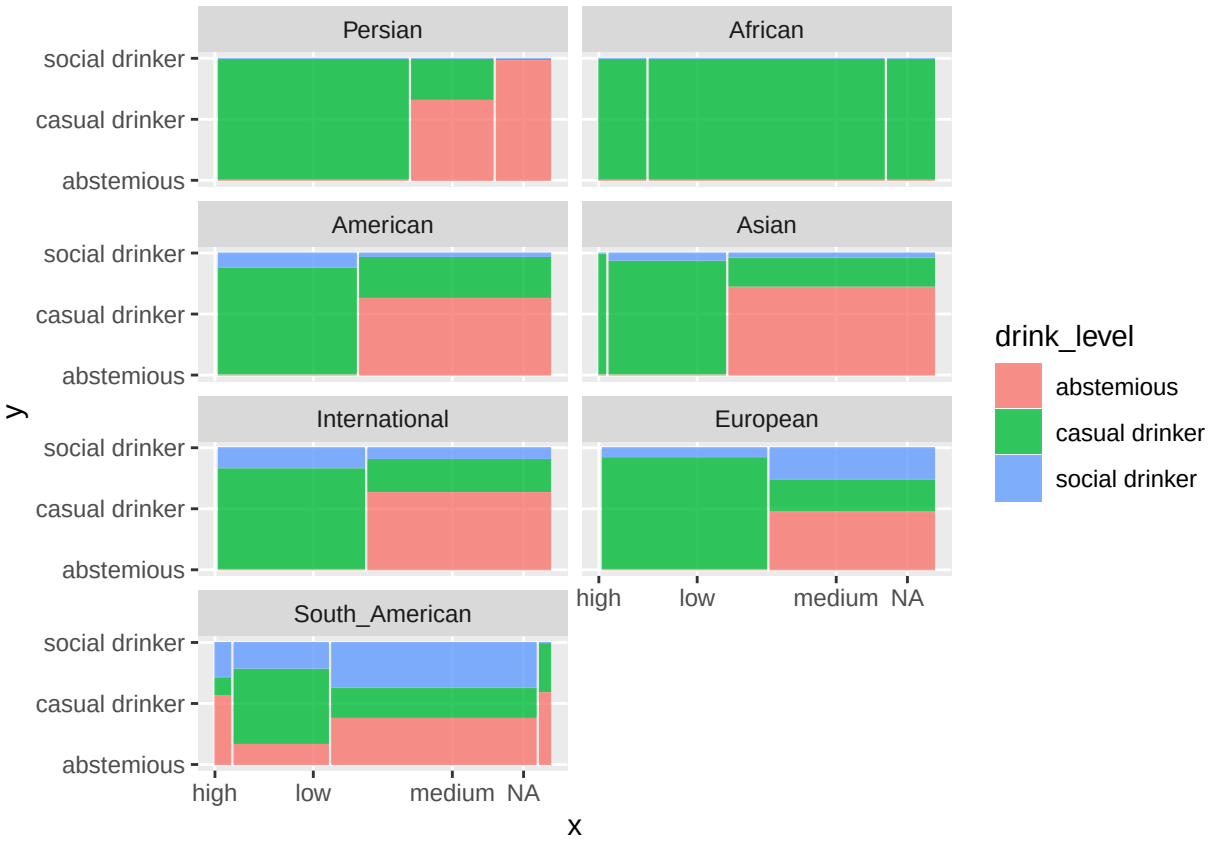



```
head(user_detail)
```

```
##   userID latitude longitude smoker    drink_level dress_preference
## 1  U1001 22.14000 -100.9788  false     abstemious         informal
## 2  U1002 22.15009 -100.9833  false     abstemious         informal
## 3  U1003 22.11985 -100.9465  false social drinker           formal
## 4  U1004 18.86700  -99.1830  false     abstemious         informal
## 5  U1004 18.86700  -99.1830  false     abstemious         informal
## 6  U1004 18.86700  -99.1830  false     abstemious         informal
##   ambience transport marital_status       hijos birth_year   interest
## 1   family   on foot         single independent       1989    variety
## 2   family    public         single independent       1990 technology
## 3   family    public         single independent       1989       none
## 4   family    public         single independent       1940    variety
## 5   family    public         single independent       1940    variety
## 6   family    public         single independent       1940    variety
##           personality religion     activity color weight budget height
## 1   thrifty-protector     none      student black     69 medium   1.77
## 2 hunter-ostentatious Catholic      student   red     40    low   1.87
## 3         hard-worker Catholic      student  blue     60    low   1.69
## 4         hard-worker     none professional green     44 medium   1.53
## 5         hard-worker     none professional green     44 medium   1.53
## 6         hard-worker     none professional green     44 medium   1.53
##         Rcuisine         Upayment
## 1       American             cash
## 2 South_American             cash
```

```
## 3 South_American             cash
## 4  International             cash
## 5  International bank_debit_cards
## 6  International             cash
```

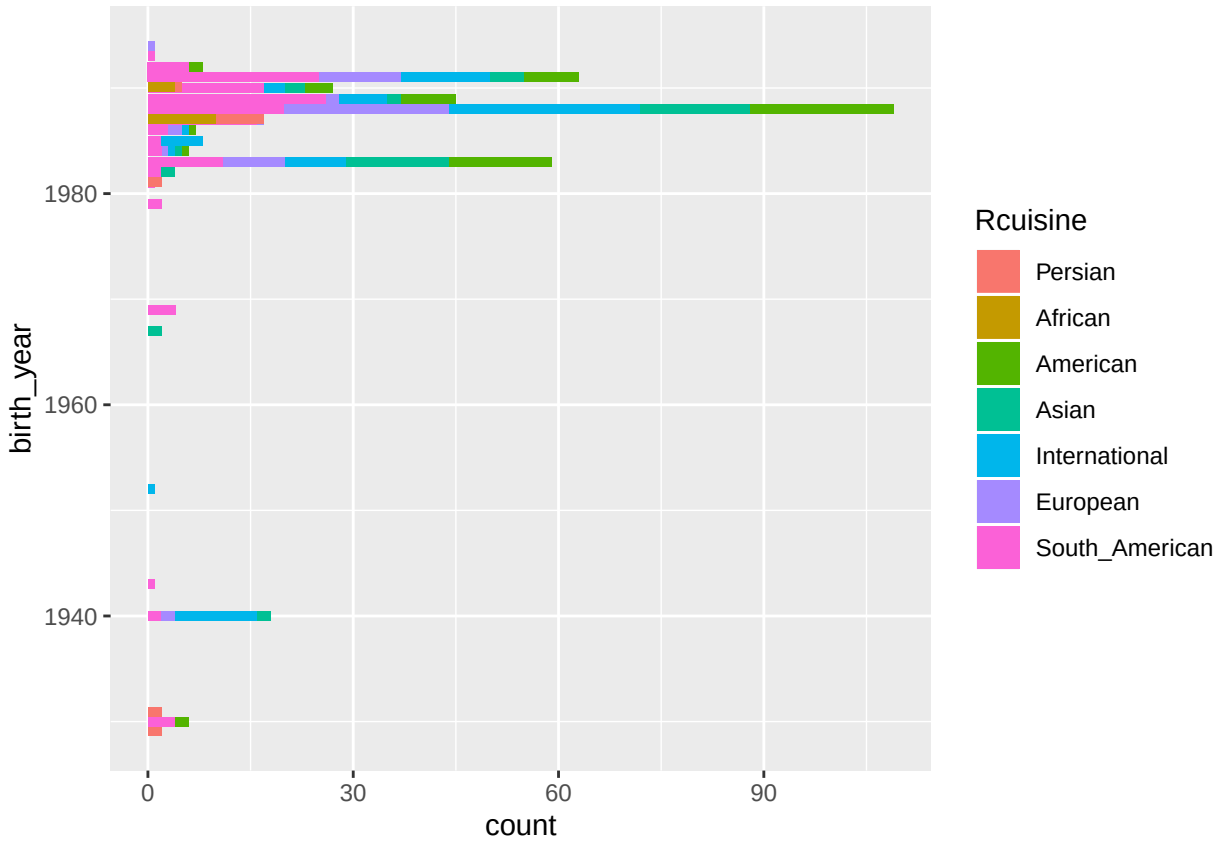
```
ggplot(user_detail) + geom_mosaic(aes(product(drink_level,budget), fill = drink_level)) + facet_wrap(.~
```



Age

```
ggplot(user_detail, aes(birth_year, fill=Rcuisine)) + geom_bar() + coord_flip()
```

```
## Warning: position_stack requires non-overlapping x intervals
```



1.5.3 Rating Data

Bewertugen der Places mit cuisine und name

```
head(rating)
```

```
##   userID placeID rating food_rating service_rating
## 1  U1077  135085      2           2              2
## 2  U1077  135038      2           2              1
## 3  U1077  132825      2           2              2
## 4  U1077  135060      1           2              2
## 5  U1068  135104      1           1              2
## 6  U1068  132740      0           0              0
```

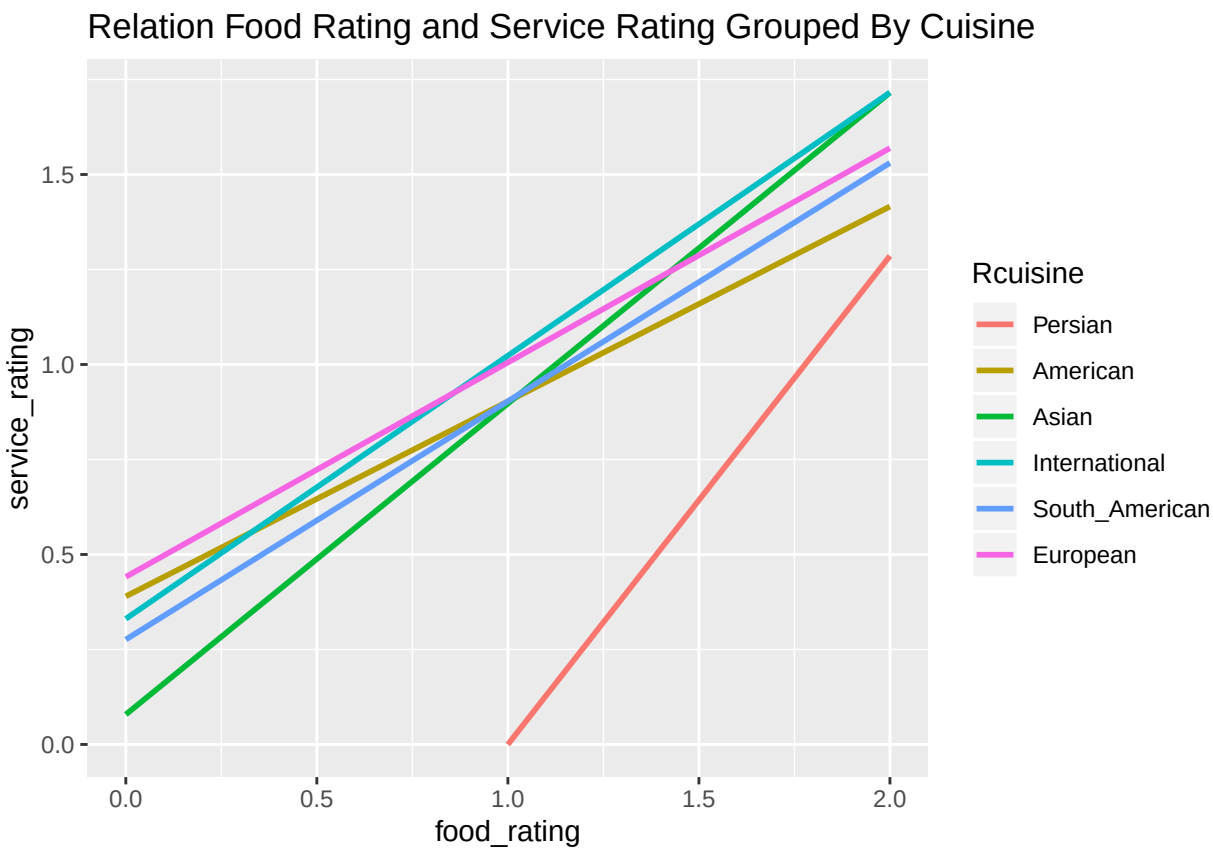
```
rating_detailed <- rating %>%
  inner_join(cuisine) %>%
  inner_join(geoplaces) %>%
  arrange(placeID) %>%
  select(placeID, rating, food_rating, service_rating, name, Rcuisine)
```

```
## Joining, by = "placeID"
## Joining, by = "placeID"
```

```
rating_detailed %>%
  group_by(placeID, name, Rcuisine) %>%
  summarize(rating = mean(rating),
            food_rating = mean(food_rating),
            service_rating = mean(food_rating))
```

```
## # A tibble: 101 x 6
## # Groups:   placeID, name [95]
##    placeID name                Rcuisine   rating food_rating service_rating
##      <int> <fct>               <fct>       <dbl>       <dbl>          <dbl>
##  1  132560 puesto de gorditas  Internati~   0.5         1              1
##  2  132572 Cafe Chaires        Internati~   1           1              1
##  3  132583 McDonalds Centro    American     1           1              1
##  4  132584 Gorditas Dona Tota  South_Ame~   1.33        1.5            1.5
##  5  132594 tacos de barbacoa ~ South_Ame~   0.6         1.2            1.2
##  6  132608 Hamburguesas La pe~ South_Ame~   1           1.17           1.17
##  7  132609 Pollo_Frito_Buenos~ American     0.6         0.6            0.6
##  8  132613 carnitas_mata       South_Ame~   1.17        1.33           1.33
##  9  132626 la perica hamburgu~ European     1.25        1              1
## 10  132630 palomo tec          South_Ame~   1.17        1.17           1.17
## # ... with 91 more rows
```

```
ggplot(rating_detailed) + aes(food_rating, service_rating, col = Rcuisine) +
  geom_smooth(method = "lm", se=F) + ggtitle("Relation Food Rating and Service Rating Grouped By Cuisine
```



Rating nach Alter und Cuisine

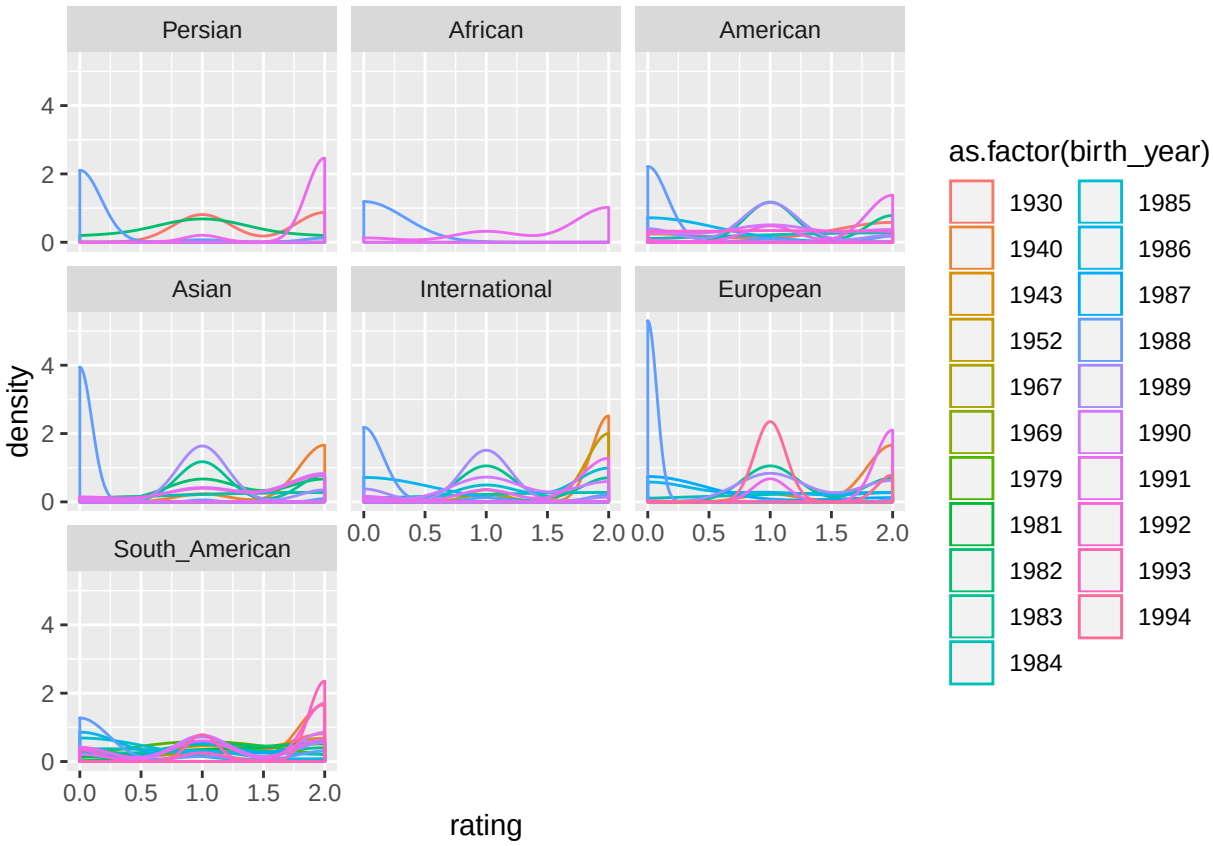
```
rating_detailed_user <- rating %>%
  join(userprofile) %>%
  join(usercuisine)
```

```
## Joining by: userID
## Joining by: userID
```

```
head(rating_detailed_user)
```

```
##   userID placeID rating food_rating service_rating latitude longitude
## 1  U1077  135085      2           2              2 22.15647 -100.9855
## 2  U1077  135038      2           2              1 22.15647 -100.9855
## 3  U1077  132825      2           2              2 22.15647 -100.9855
## 4  U1077  135060      1           2              2 22.15647 -100.9855
## 5  U1068  135104      1           1              2 23.75227  -99.1686
## 6  U1068  132740      0           0              0 23.75227  -99.1686
##   smoker    drink_level dress_preference ambience transport marital_status
## 1  false social drinker          elegant   family    public        married
## 2  false social drinker          elegant   family    public        married
## 3  false social drinker          elegant   family    public        married
## 4  false social drinker          elegant   family    public        married
## 5  false casual drinker         informal  friends    public         single
## 6  false casual drinker         informal  friends    public         single
##         hijos birth_year   interest       personality religion activity
## 1        kids       1987 technology thrifty-protector Catholic  student
## 2        kids       1987 technology thrifty-protector Catholic  student
## 3        kids       1987 technology thrifty-protector Catholic  student
## 4        kids       1987 technology thrifty-protector Catholic  student
## 5 independent       1988 technology thrifty-protector Catholic  student
## 6 independent       1988 technology thrifty-protector Catholic  student
##   color weight budget height       Rcuisine
## 1  blue     65 medium   1.71 South_American
## 2  blue     65 medium   1.71 South_American
## 3  blue     65 medium   1.71 South_American
## 4  blue     65 medium   1.71 South_American
## 5  blue     72    low   1.57 South_American
## 6  blue     72    low   1.57 South_American
```

```
ggplot(rating_detailed_user, aes(rating, colour = as.factor(birth_year))) + geom_density() + facet_wrap
```



Rating nach Preis Klasse

```
rating_detailed_price <- rating %>%
  join(cuisine) %>%
  join(geoplaces)
```

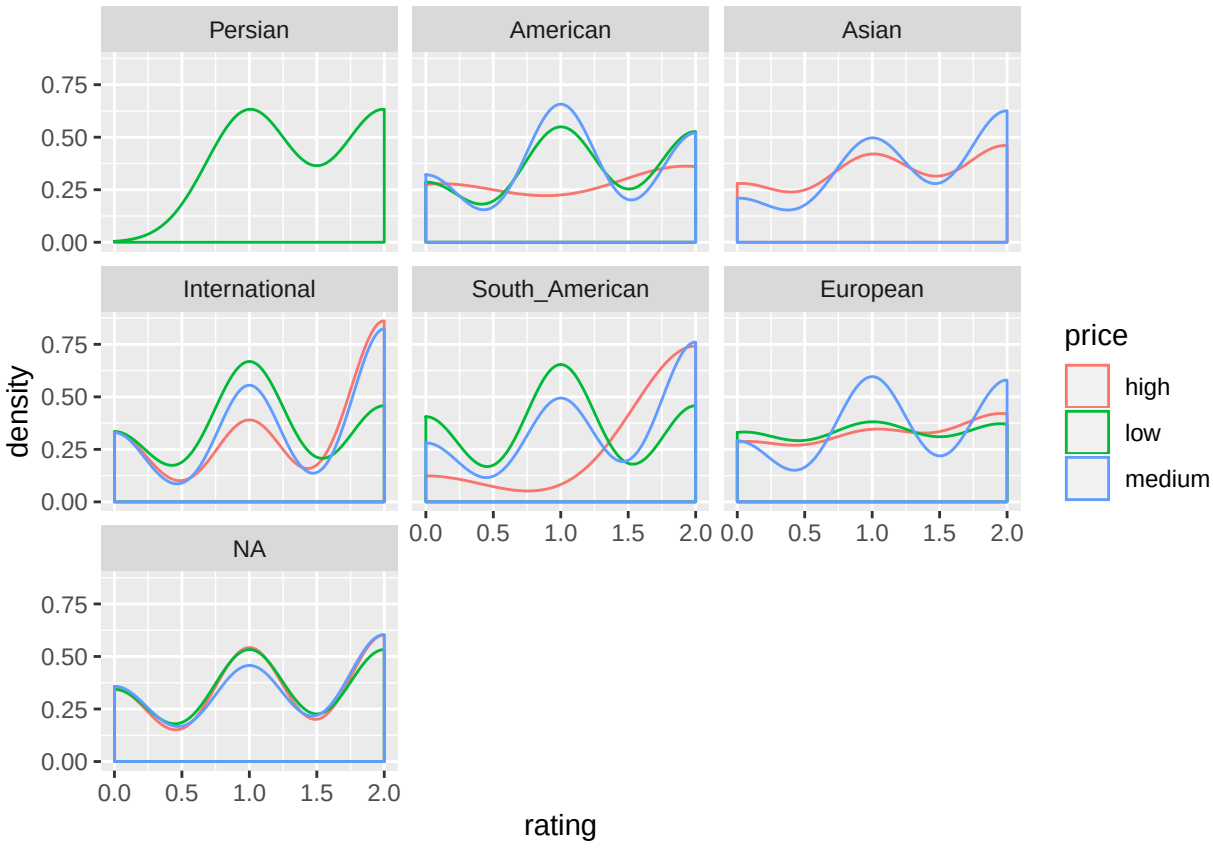
```
## Joining by: placeID
## Joining by: placeID
```

```
head(rating_detailed_price)
```

```
##   userID placeID rating food_rating service_rating       Rcuisine latitude
## 1  U1077  135085      2           2              2       American 22.15080
## 2  U1077  135038      2           2              1           <NA> 22.15565
## 3  U1077  132825      2           2              2 South_American 22.14739
## 4  U1077  135060      1           2              2       European 22.15688
## 5  U1068  135104      1           1              2 South_American 23.75298
## 6  U1068  132740      0           0              0 South_American 23.75220
##    longitude                                    the_geom_meter
## 1 -100.98268 0101000020957F00009F823DA6094858C18A2D4D37F9A44B41
## 2 -100.97777 0101000020957F0000506149736E4758C1A8BC93DA48A34B41
## 3 -100.98309 0101000020957F00001AD016568C4858C1243261274BA54B41
## 4 -100.97849 0101000020957F00004C95C918394758C17A5C44896AA34B41
## 5  -99.16843 0101000020957F00007CDF5EAFC58157C1645743B23E4F4941
## 6  -99.16663 0101000020957F000027A30471EE8157C1AC17D61EC84E4941
##                          name                                address
```

```
## 1        Tortas Locas Hipocampo         Venustiano Carranza 719 Centro
## 2         Restaurant la Chalita     Guajardo Sn San Luis Potosi Centro
## 3               puesto de tacos esquina santos degollado y leon guzman
## 4       Restaurante Marisco Sam             Ignacio Allende 785 Centro
## 5                          vips                                   <NA>
## 6 Carreton de Flautas y Migadas                                   <NA>
##              city      state country fax   zip           alcohol
## 1 San Luis Potosi        SLP  Mexico  NA 78000 No_Alcohol_Served
## 2 San Luis Potosi        SLP  Mexico  NA 78000 No_Alcohol_Served
## 3          s.l.p.     s.l.p.  mexico  NA 78280 No_Alcohol_Served
## 4 San Luis Potosi        SLP  Mexico  NA 78310 No_Alcohol_Served
## 5            <NA>       <NA>    <NA>  NA  <NA>          Full_Bar
## 6 Ciudad Victoria Tamaulipas  Mexico  NA  <NA> No_Alcohol_Served
##    smoking_area dress_code    accessibility  price  url Rambience
## 1 not permitted   informal no_accessibility medium <NA>  familiar
## 2       section   informal no_accessibility medium <NA>  familiar
## 3          none   informal       completely    low <NA>  familiar
## 4          none   informal no_accessibility medium <NA>  familiar
## 5 not permitted   informal       completely medium <NA>  familiar
## 6     permitted   informal       completely    low <NA>  familiar
##   franchise   area other_services
## 1         f closed           none
## 2         f closed           none
## 3         f   open           none
## 4         f closed           none
## 5         t closed        variety
## 6         f   open           none
```

```
ggplot(rating_detailed_price, aes(rating, colour = price)) + geom_density() + facet_wrap(.~Rcuisine)
```



1.6 Modellierung (Klassifikation oder Regression) mit zumindest 3 Methoden, inkl. Parameter Tuning und Benchmarking [30%]

Für die Modelle wird das Rating vorhergesagt aufgrund der Prädiktoren Cuisine, smoker, budget, drinking_level, birth_year. Als Methoden wurden Rpart, RandomForest, NaiveBayes, KNN und NNET gewählt, die miteinander verglichen werden. (User-Daten)

Für die Modelle wird die Cuisine vorhergesagt aufgrund der Prädiktoren smoker, drink_level, budget, personality vom User Profile. Als Methoden wurden Rpart, RandomForest, NaiveBayes und NNET gewählt, die miteinander verglichen werden.

Durch diese zwei Modelle werden somit zwei verschiedene Werte vorhergesagt. Ein Modell sagt das Rating vor und das zweite die Cuisine.

1.6.1 Vorbereitung der Daten

Data Values

```
## Rating Data
rating_user_detail <- rating %>%
  join(userprofile) %>%
  join(cuisine)
```

```
## Joining by: userID
```

```
## Joining by: placeID
```

```r
data_rating <- rating_user_detail %>%
  mutate(rating = as.factor(rating)) %>%
  select(rating , smoker, drink_level, budget, birth_year, Rcuisine) # without food_rating, service_rat

## Cuisine Data
data_cuisine <- userprofile %>%
  join(usercuisine) %>%
  select(smoker, drink_level, budget, personality, Rcuisine)
```

```
## Joining by: userID
```

```r
head(data_rating)
```

```
##   rating smoker    drink_level budget birth_year       Rcuisine
## 1      2  false social drinker medium       1987       American
## 2      2  false social drinker medium       1987           <NA>
## 3      2  false social drinker medium       1987 South_American
## 4      1  false social drinker medium       1987       European
## 5      1  false casual drinker    low       1988 South_American
## 6      0  false casual drinker    low       1988 South_American
```

```r
head(data_cuisine)
```

```
##   smoker    drink_level budget         personality       Rcuisine
## 1  false     abstemious medium   thrifty-protector       American
## 2  false     abstemious    low hunter-ostentatious South_American
## 3  false social drinker    low         hard-worker South_American
## 4  false     abstemious medium         hard-worker  International
## 5  false     abstemious medium         hard-worker  International
## 6  false     abstemious medium         hard-worker          Asian
```

Splitting

```r
set.seed(4711)

## Rating Data
N = nrow(data_rating)
train_ind = sample(1: N, size = N * 2 / 3)

train_rating = data_rating[train_ind,]
test_rating = data_rating[-train_ind,]

## Cuisine Data
N = nrow(data_cuisine)
train_ind = sample(1: N, size = N * 2 / 3)

train_cuisine = data_cuisine[train_ind,]
test_cuisine = data_cuisine[-train_ind,]
```

Scaling (with Encoding)

```
## Rating Data
scaler = preProcess(train_rating)
train_rating_scaled = predict(scaler, train_rating)
test_rating_scaled = predict(scaler, test_rating)

encoder = dummyVars( ~ rating + smoker + drink_level + budget + birth_year + Rcuisine, data = train_rati
encoded_train_rating = as.data.frame(predict(encoder,train_rating_scaled))
encoded_test_rating = as.data.frame(predict(encoder,test_rating_scaled))

## Cuisine Data
scaler = preProcess(train_cuisine)
```

```
## Warning in pre_process_options(method, column_types): The following pre-
## processing methods were eliminated: 'center', 'scale'
```

```
train_cuisine_scaled = predict(scaler, train_cuisine)
test_cuisine_scaled = predict(scaler, test_cuisine)

encoder = dummyVars( ~ Rcuisine + smoker + drink_level + budget + personality, data = train_cuisine_scal
encoded_train_cuisine = as.data.frame(predict(encoder,train_cuisine_scaled))
encoded_test_cuisine = as.data.frame(predict(encoder,test_cuisine_scaled))

# Removing NA
train_rating_scaled_na_free <- na.omit(train_rating_scaled)
train_cuisine_scaled_na_free <- na.omit(train_cuisine_scaled)

test_rating_scaled_na_free <- na.omit(test_rating_scaled)
test_cuisine_scaled_na_free <- na.omit(test_cuisine_scaled)

# Rating KNN prep
df = data_frame(rating = train_rating_scaled_na_free[,1], smoker = as.matrix(class2ind(train_rating_scal
                drink_level = as.matrix(class2ind(train_rating_scaled_na_free[,3])),
                budget = as.matrix(class2ind(train_rating_scaled_na_free[,4])),
                birth_year = train_rating_scaled_na_free$birth_year,
                Rcuisine = as.matrix(class2ind(train_rating_scaled_na_free$Rcuisine)))
```

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

```
df_test = data_frame(rating = test_rating_scaled_na_free[,1], smoker = as.matrix(class2ind(test_rating_s
                drink_level = as.matrix(class2ind(test_rating_scaled_na_free[,3])),
                budget = as.matrix(class2ind(test_rating_scaled_na_free[,4])),
                birth_year = test_rating_scaled_na_free$birth_year,
                Rcuisine = as.matrix(class2ind(test_rating_scaled_na_free$Rcuisine)))
```

1.6.2 Tunen der Modelle

1.6.2.1 Tunen für Rating - Prediction

KNN:

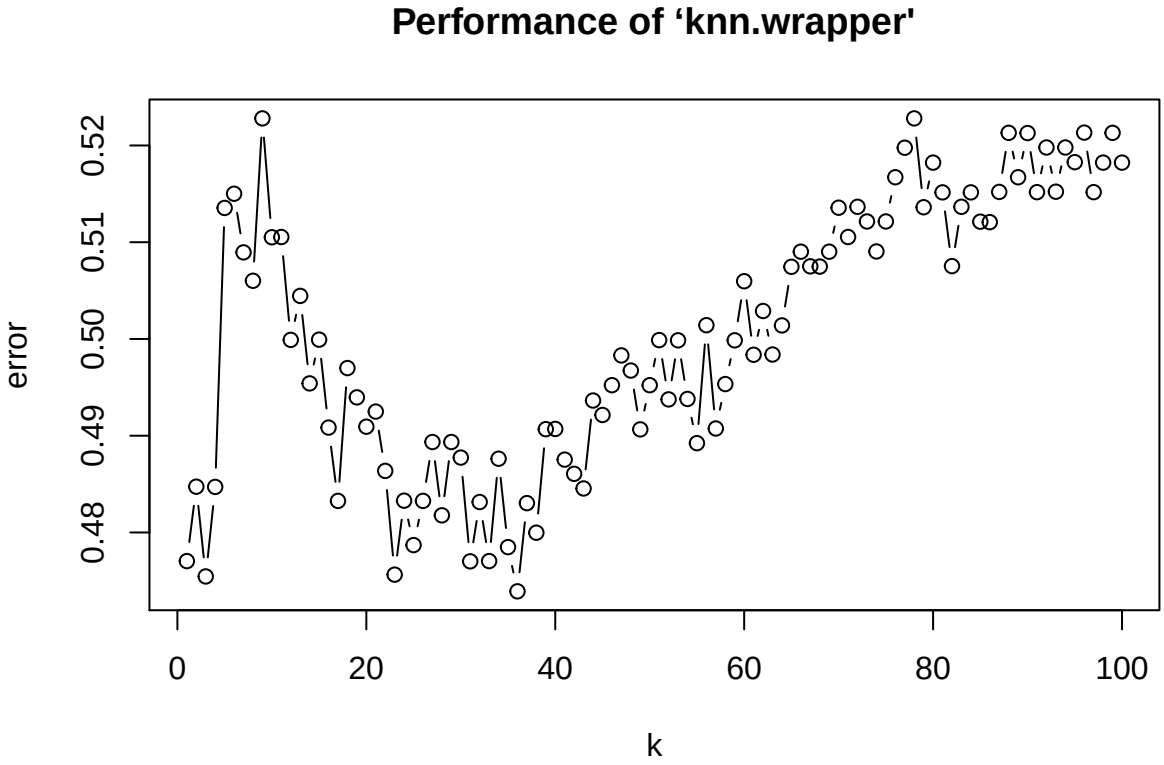
```r
tunobj_knn = tune.knn(df[,-1], df$rating,
                      na.action = na.omit,
                      k = 1 : 100,
                      tunecontrol = tune.control(sampling = "cross",cross = 10))
summary(tunobj_knn)
```

```
##
## Parameter tuning of 'knn.wrapper':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##   k
##  36
##
## - best performance: 0.4739161
##
## - Detailed performance results:
##       k     error dispersion
## 1     1 0.4770396 0.02684158
## 2     2 0.4847319 0.04924043
## 3     3 0.4754545 0.05138708
## 4     4 0.4847086 0.06321084
## 5     5 0.5135431 0.05756285
## 6     6 0.5150117 0.07611651
## 7     7 0.5089510 0.07130685
## 8     8 0.5060140 0.06430664
## 9     9 0.5227972 0.06263820
## 10   10 0.5105128 0.07294109
## 11   11 0.5105361 0.07106578
## 12   12 0.4999068 0.07352967
## 13   13 0.5044522 0.08451415
## 14   14 0.4954079 0.06199230
## 15   15 0.4999301 0.06503771
## 16   16 0.4908392 0.06494714
## 17   17 0.4832634 0.06780771
## 18   18 0.4969930 0.06620270
## 19   19 0.4939627 0.06841477
## 20   20 0.4909324 0.06721757
## 21   21 0.4924942 0.07161628
## 22   22 0.4863636 0.07223302
## 23   23 0.4756410 0.06828544
## 24   24 0.4832867 0.06186596
## 25   25 0.4786946 0.06510814
## 26   26 0.4832634 0.07236959
## 27   27 0.4893473 0.06685233
## 28   28 0.4817716 0.06323085
## 29   29 0.4893473 0.06727984
## 30   30 0.4877389 0.07172085
## 31   31 0.4770163 0.06257507
## 32   32 0.4831469 0.06116137
## 33   33 0.4770396 0.06862322
## 34   34 0.4876224 0.07229594
```

```
## 35   35 0.4784848 0.06344154
## 36   36 0.4739161 0.07085348
## 37   37 0.4830303 0.07229555
## 38   38 0.4799767 0.07580720
## 39   39 0.4906760 0.07740932
## 40   40 0.4906993 0.06994780
## 41   41 0.4875291 0.07096397
## 42   42 0.4860606 0.06624740
## 43   43 0.4845455 0.06614993
## 44   44 0.4936364 0.07563144
## 45   45 0.4921445 0.07015573
## 46   46 0.4952214 0.06889647
## 47   47 0.4982984 0.07400516
## 48   48 0.4967366 0.07027670
## 49   49 0.4906527 0.06986170
## 50   50 0.4952214 0.06885053
## 51   51 0.4998834 0.06819874
## 52   52 0.4937529 0.06622735
## 53   53 0.4998601 0.06917316
## 54   54 0.4937995 0.06892126
## 55   55 0.4892308 0.06983943
## 56   56 0.5014219 0.07126783
## 57   57 0.4907459 0.06990944
## 58   58 0.4953380 0.06688564
## 59   59 0.4998601 0.06516060
## 60   60 0.5059674 0.06523644
## 61   61 0.4983683 0.06780052
## 62   62 0.5028904 0.07079097
## 63   63 0.4983916 0.07100490
## 64   64 0.5013986 0.06822784
## 65   65 0.5074592 0.06588105
## 66   66 0.5090210 0.06226158
## 67   67 0.5075058 0.06932953
## 68   68 0.5074825 0.06365087
## 69   69 0.5090210 0.06582156
## 70   70 0.5135664 0.06977638
## 71   71 0.5105361 0.06916789
## 72   72 0.5136597 0.07467913
## 73   73 0.5121445 0.07120757
## 74   74 0.5090443 0.06788113
## 75   75 0.5121445 0.07005699
## 76   76 0.5167133 0.07230860
## 77   77 0.5197669 0.06752488
## 78   78 0.5227972 0.06611814
## 79   79 0.5136131 0.06960741
## 80   80 0.5182284 0.06918884
## 81   81 0.5151515 0.06772898
## 82   82 0.5075291 0.07366468
## 83   83 0.5136597 0.07254635
## 84   84 0.5151515 0.07363721
## 85   85 0.5121212 0.08270829
## 86   86 0.5120746 0.08111060
## 87   87 0.5151981 0.07590886
## 88   88 0.5213054 0.07766924
```

```
## 89   89 0.5167133 0.07703790
## 90   90 0.5212821 0.07912028
## 91   91 0.5151748 0.08035168
## 92   92 0.5197902 0.08480681
## 93   93 0.5152214 0.08585750
## 94   94 0.5197902 0.08395618
## 95   95 0.5182751 0.08415741
## 96   96 0.5213287 0.08822427
## 97   97 0.5151748 0.09129329
## 98   98 0.5182284 0.08127880
## 99   99 0.5213054 0.08475680
## 100 100 0.5182284 0.08342773
```

```
plot(tunobj_knn)
```



Für das KNN Modell wird k=36 gewählt.

RPART:

```
tuneobj_rpart = tune.rpart(rating ~ smoker + drink_level + budget + birth_year + Rcuisine, data = train_
                           na.action = na.omit,
                           minsplit = 1:100,
                           tunecontrol = tune.control(sampling = "fix", cross = N, nrepeat=10))

summary(tuneobj_rpart)
```

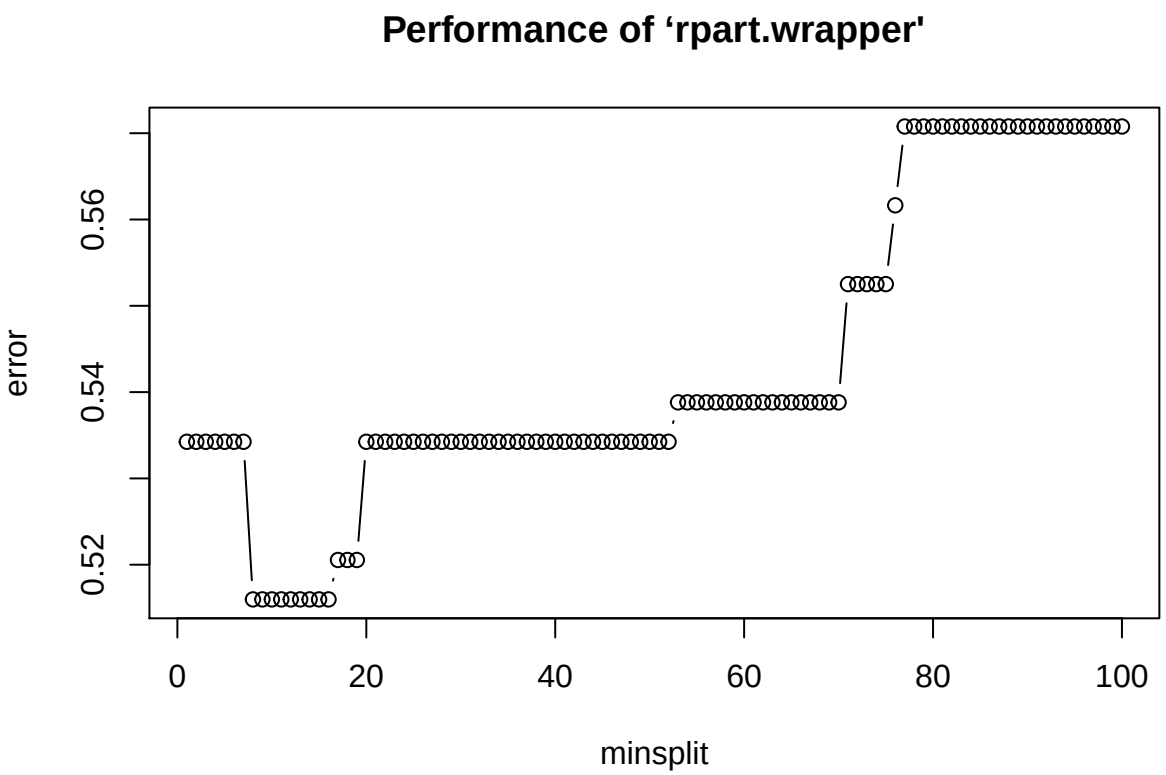
```
##
## Parameter tuning of 'rpart.wrapper':
##
```

```
## - sampling method: fixed training/validation set
##
## - best parameters:
##  minsplit
##         8
##
## - best performance: 0.5159817
##
## - Detailed performance results:
##     minsplit     error dispersion
## 1          1 0.5342466         NA
## 2          2 0.5342466         NA
## 3          3 0.5342466         NA
## 4          4 0.5342466         NA
## 5          5 0.5342466         NA
## 6          6 0.5342466         NA
## 7          7 0.5342466         NA
## 8          8 0.5159817         NA
## 9          9 0.5159817         NA
## 10        10 0.5159817         NA
## 11        11 0.5159817         NA
## 12        12 0.5159817         NA
## 13        13 0.5159817         NA
## 14        14 0.5159817         NA
## 15        15 0.5159817         NA
## 16        16 0.5159817         NA
## 17        17 0.5205479         NA
## 18        18 0.5205479         NA
## 19        19 0.5205479         NA
## 20        20 0.5342466         NA
## 21        21 0.5342466         NA
## 22        22 0.5342466         NA
## 23        23 0.5342466         NA
## 24        24 0.5342466         NA
## 25        25 0.5342466         NA
## 26        26 0.5342466         NA
## 27        27 0.5342466         NA
## 28        28 0.5342466         NA
## 29        29 0.5342466         NA
## 30        30 0.5342466         NA
## 31        31 0.5342466         NA
## 32        32 0.5342466         NA
## 33        33 0.5342466         NA
## 34        34 0.5342466         NA
## 35        35 0.5342466         NA
## 36        36 0.5342466         NA
## 37        37 0.5342466         NA
## 38        38 0.5342466         NA
## 39        39 0.5342466         NA
## 40        40 0.5342466         NA
## 41        41 0.5342466         NA
## 42        42 0.5342466         NA
## 43        43 0.5342466         NA
## 44        44 0.5342466         NA
```

```
## 45        45 0.5342466         NA
## 46        46 0.5342466         NA
## 47        47 0.5342466         NA
## 48        48 0.5342466         NA
## 49        49 0.5342466         NA
## 50        50 0.5342466         NA
## 51        51 0.5342466         NA
## 52        52 0.5342466         NA
## 53        53 0.5388128         NA
## 54        54 0.5388128         NA
## 55        55 0.5388128         NA
## 56        56 0.5388128         NA
## 57        57 0.5388128         NA
## 58        58 0.5388128         NA
## 59        59 0.5388128         NA
## 60        60 0.5388128         NA
## 61        61 0.5388128         NA
## 62        62 0.5388128         NA
## 63        63 0.5388128         NA
## 64        64 0.5388128         NA
## 65        65 0.5388128         NA
## 66        66 0.5388128         NA
## 67        67 0.5388128         NA
## 68        68 0.5388128         NA
## 69        69 0.5388128         NA
## 70        70 0.5388128         NA
## 71        71 0.5525114         NA
## 72        72 0.5525114         NA
## 73        73 0.5525114         NA
## 74        74 0.5525114         NA
## 75        75 0.5525114         NA
## 76        76 0.5616438         NA
## 77        77 0.5707763         NA
## 78        78 0.5707763         NA
## 79        79 0.5707763         NA
## 80        80 0.5707763         NA
## 81        81 0.5707763         NA
## 82        82 0.5707763         NA
## 83        83 0.5707763         NA
## 84        84 0.5707763         NA
## 85        85 0.5707763         NA
## 86        86 0.5707763         NA
## 87        87 0.5707763         NA
## 88        88 0.5707763         NA
## 89        89 0.5707763         NA
## 90        90 0.5707763         NA
## 91        91 0.5707763         NA
## 92        92 0.5707763         NA
## 93        93 0.5707763         NA
## 94        94 0.5707763         NA
## 95        95 0.5707763         NA
## 96        96 0.5707763         NA
## 97        97 0.5707763         NA
## 98        98 0.5707763         NA
```

```
## 99        99 0.5707763         NA
## 100      100 0.5707763         NA
```

```
plot(tuneobj_rpart)
```



Minsplit wird auf 47 gesetzt beim Rpart-Modell

RandomForest:

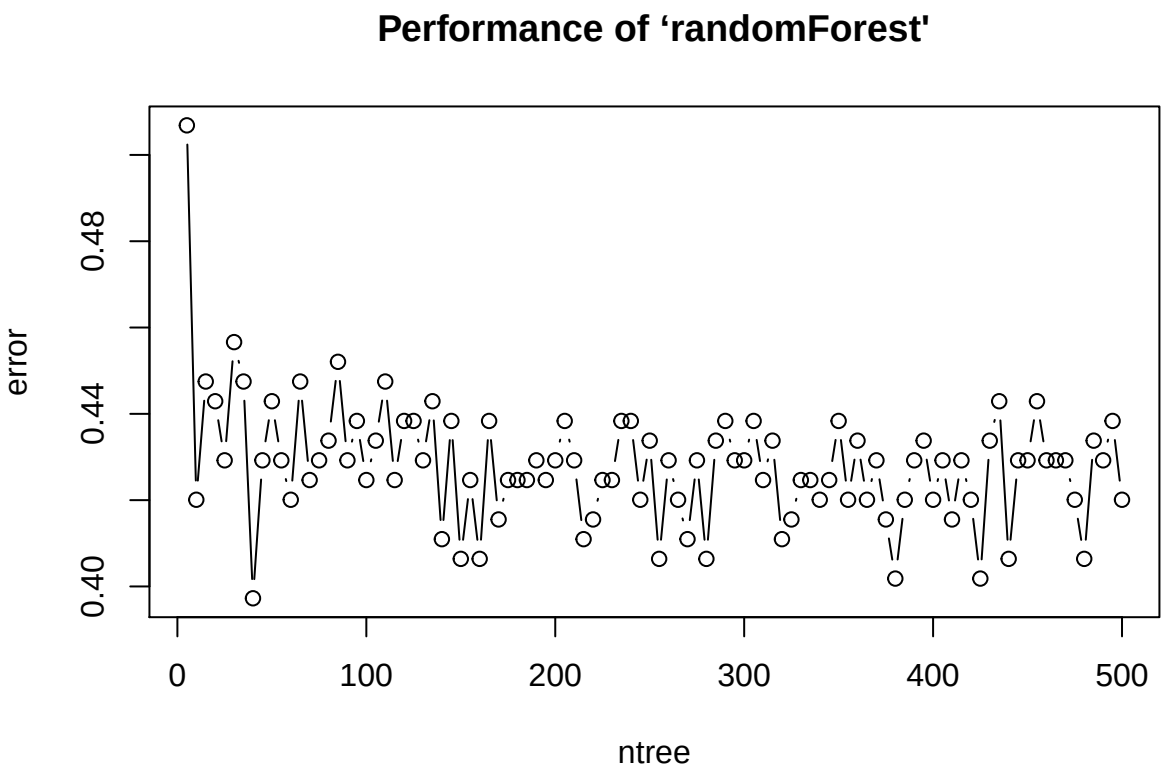
```
tuneobj_rf = tune.randomForest(rating ~ smoker + drink_level + budget + birth_year + Rcuisine, data = t
                               ntree = 1:100 * 5,
                               tunecontrol = tune.control(sampling = "fix"))

summary(tuneobj_rf)
```

```
##
## Parameter tuning of 'randomForest':
##
## - sampling method: fixed training/validation set
##
## - best parameters:
##  ntree
##     40
##
## - best performance: 0.3972603
##
## - Detailed performance results:
##    ntree     error dispersion
## 1      5 0.5068493         NA
```

```
## 2      10 0.4200913         NA
## 3      15 0.4474886         NA
## 4      20 0.4429224         NA
## 5      25 0.4292237         NA
## 6      30 0.4566210         NA
## 7      35 0.4474886         NA
## 8      40 0.3972603         NA
## 9      45 0.4292237         NA
## 10     50 0.4429224         NA
## 11     55 0.4292237         NA
## 12     60 0.4200913         NA
## 13     65 0.4474886         NA
## 14     70 0.4246575         NA
## 15     75 0.4292237         NA
## 16     80 0.4337900         NA
## 17     85 0.4520548         NA
## 18     90 0.4292237         NA
## 19     95 0.4383562         NA
## 20    100 0.4246575         NA
## 21    105 0.4337900         NA
## 22    110 0.4474886         NA
## 23    115 0.4246575         NA
## 24    120 0.4383562         NA
## 25    125 0.4383562         NA
## 26    130 0.4292237         NA
## 27    135 0.4429224         NA
## 28    140 0.4109589         NA
## 29    145 0.4383562         NA
## 30    150 0.4063927         NA
## 31    155 0.4246575         NA
## 32    160 0.4063927         NA
## 33    165 0.4383562         NA
## 34    170 0.4155251         NA
## 35    175 0.4246575         NA
## 36    180 0.4246575         NA
## 37    185 0.4246575         NA
## 38    190 0.4292237         NA
## 39    195 0.4246575         NA
## 40    200 0.4292237         NA
## 41    205 0.4383562         NA
## 42    210 0.4292237         NA
## 43    215 0.4109589         NA
## 44    220 0.4155251         NA
## 45    225 0.4246575         NA
## 46    230 0.4246575         NA
## 47    235 0.4383562         NA
## 48    240 0.4383562         NA
## 49    245 0.4200913         NA
## 50    250 0.4337900         NA
## 51    255 0.4063927         NA
## 52    260 0.4292237         NA
## 53    265 0.4200913         NA
## 54    270 0.4109589         NA
## 55    275 0.4292237         NA
```

```
## 56    280 0.4063927         NA
## 57    285 0.4337900         NA
## 58    290 0.4383562         NA
## 59    295 0.4292237         NA
## 60    300 0.4292237         NA
## 61    305 0.4383562         NA
## 62    310 0.4246575         NA
## 63    315 0.4337900         NA
## 64    320 0.4109589         NA
## 65    325 0.4155251         NA
## 66    330 0.4246575         NA
## 67    335 0.4246575         NA
## 68    340 0.4200913         NA
## 69    345 0.4246575         NA
## 70    350 0.4383562         NA
## 71    355 0.4200913         NA
## 72    360 0.4337900         NA
## 73    365 0.4200913         NA
## 74    370 0.4292237         NA
## 75    375 0.4155251         NA
## 76    380 0.4018265         NA
## 77    385 0.4200913         NA
## 78    390 0.4292237         NA
## 79    395 0.4337900         NA
## 80    400 0.4200913         NA
## 81    405 0.4292237         NA
## 82    410 0.4155251         NA
## 83    415 0.4292237         NA
## 84    420 0.4200913         NA
## 85    425 0.4018265         NA
## 86    430 0.4337900         NA
## 87    435 0.4429224         NA
## 88    440 0.4063927         NA
## 89    445 0.4292237         NA
## 90    450 0.4292237         NA
## 91    455 0.4429224         NA
## 92    460 0.4292237         NA
## 93    465 0.4292237         NA
## 94    470 0.4292237         NA
## 95    475 0.4200913         NA
## 96    480 0.4063927         NA
## 97    485 0.4337900         NA
## 98    490 0.4292237         NA
## 99    495 0.4383562         NA
## 100   500 0.4200913         NA
```

```
plot(tuneobj_rf)
```



Für das RandomForest-Modell werden 50 Bäume genommen.

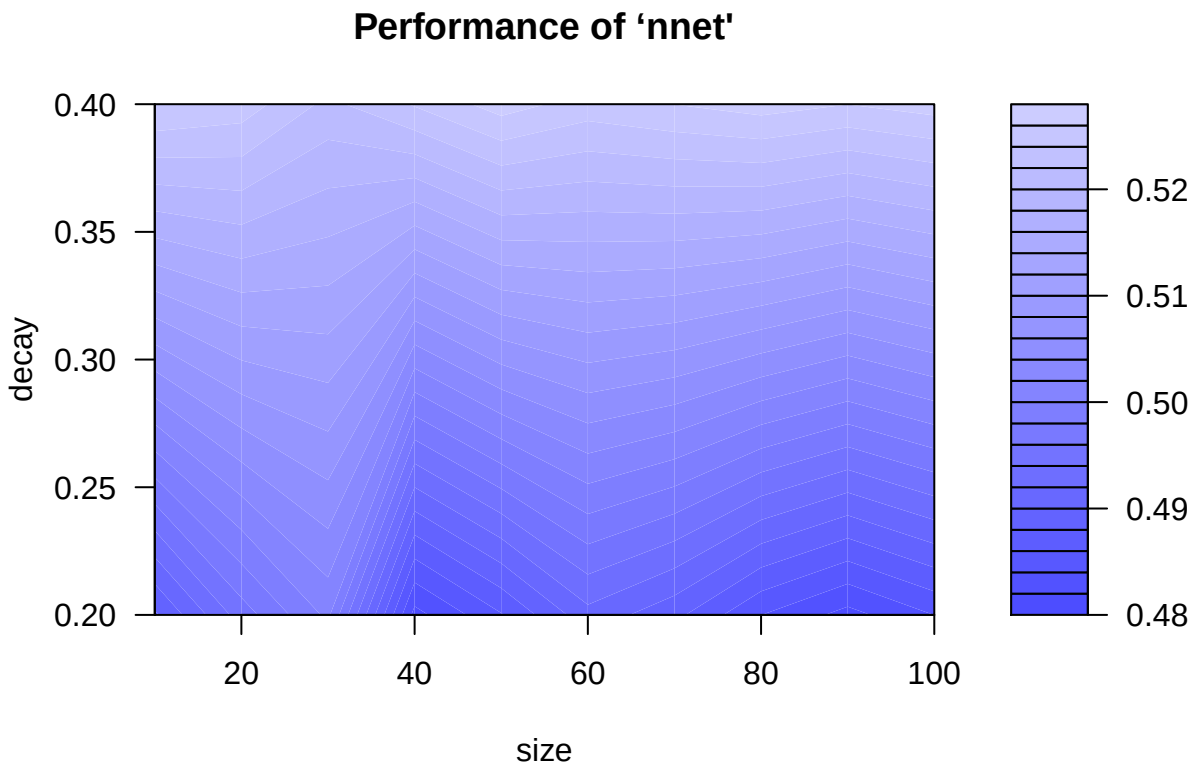
NNET:

```
tuneobj_nnet = tune.nnet(rating ~ ., data = train_rating_scaled_na_free, linout = TRUE,
                     size = 1:10 * 10, # c(10,20,30) | 1: 10 * 10
                     decay = c(0.2,0.4),
                     MaxNWts = 10000,
                     tunecontrol = tune.control(sampling = "fix", nrepeat = 5))

summary(tuneobj_nnet)
```

```
##
## Parameter tuning of 'nnet':
##
## - sampling method: fixed training/validation set
##
## - best parameters:
##  size decay
##    40   0.2
##
## - best performance: 0.4812785
##
## - Detailed performance results:
##    size decay     error dispersion
## 1    10   0.2 0.4876712         NA
## 2    20   0.2 0.4949772         NA
## 3    30   0.2 0.5004566         NA
## 4    40   0.2 0.4812785         NA
## 5    50   0.2 0.4858447         NA
```

```
## 6    60   0.2 0.4913242         NA
## 7    70   0.2 0.4885845         NA
## 8    80   0.2 0.4840183         NA
## 9    90   0.2 0.4812785         NA
## 10  100   0.2 0.4840183         NA
## 11   10   0.4 0.5260274         NA
## 12   20   0.4 0.5251142         NA
## 13   30   0.4 0.5214612         NA
## 14   40   0.4 0.5242009         NA
## 15   50   0.4 0.5269406         NA
## 16   60   0.4 0.5251142         NA
## 17   70   0.4 0.5260274         NA
## 18   80   0.4 0.5269406         NA
## 19   90   0.4 0.5260274         NA
## 20  100   0.4 0.5269406         NA
```

```
plot(tuneobj_nnet)
```



Für das NNET-Modell werden size = 10 und decay = 0.2 ausgewählt.

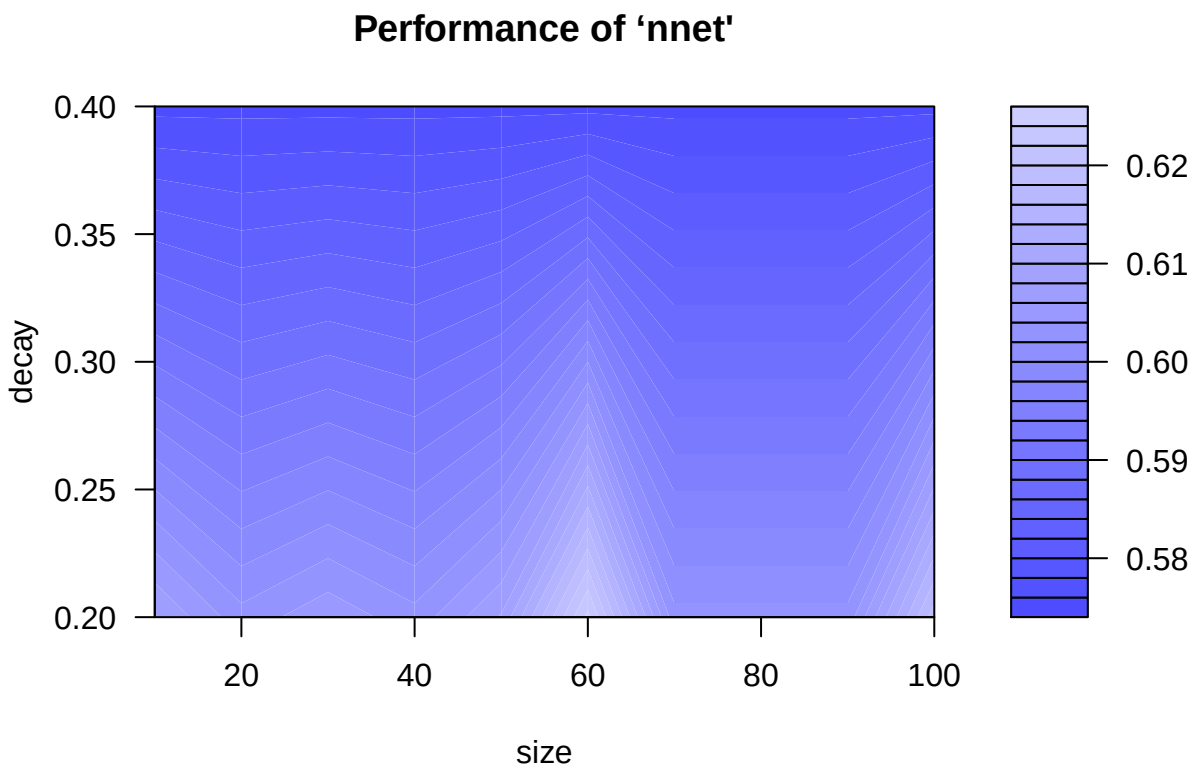
1.6.2.2 Tunen für Cuisine - Prediction

NNET:

```
tuneobj_nnet = tune.nnet(Rcuisine ~ ., data = train_cuisine_scaled_na_free, linout = TRUE,
                    size = 1:10 * 10, # c(10,20,30) | 1: 10 * 10
                    decay = c(0.2,0.4),
                    MaxNWts = 10000,
                    tunecontrol = tune.control(sampling = "fix", nrepeat = 5))
summary(tuneobj_nnet)
```

```
##
## Parameter tuning of 'nnet':
##
## - sampling method: fixed training/validation set
##
## - best parameters:
##  size decay
##    10   0.4
##
## - best performance: 0.5753425
##
## - Detailed performance results:
##    size decay     error dispersion
## 1    10   0.2 0.6082192         NA
## 2    20   0.2 0.6027397         NA
## 3    30   0.2 0.6054795         NA
## 4    40   0.2 0.6027397         NA
## 5    50   0.2 0.6082192         NA
## 6    60   0.2 0.6246575         NA
## 7    70   0.2 0.6027397         NA
## 8    80   0.2 0.6027397         NA
## 9    90   0.2 0.6027397         NA
## 10  100   0.2 0.6191781         NA
## 11   10   0.4 0.5753425         NA
## 12   20   0.4 0.5753425         NA
## 13   30   0.4 0.5753425         NA
## 14   40   0.4 0.5753425         NA
## 15   50   0.4 0.5753425         NA
## 16   60   0.4 0.5753425         NA
## 17   70   0.4 0.5753425         NA
## 18   80   0.4 0.5753425         NA
## 19   90   0.4 0.5753425         NA
## 20  100   0.4 0.5753425         NA
```

```
plot(tuneobj_nnet)
```



Für das NNET-Modell werden size = 30 und decay = 0.2 ausgewählt.

RPART:

```
tuneobj_rpart = tune.rpart(Rcuisine ~ smoker + drink_level + budget + personality, data = train_cuisine_
                           na.action = na.omit,
                           minsplit = 1:150,
                           tunecontrol = tune.control(sampling = "fix", cross = N, nrepeat=5))

summary(tuneobj_rpart)
```

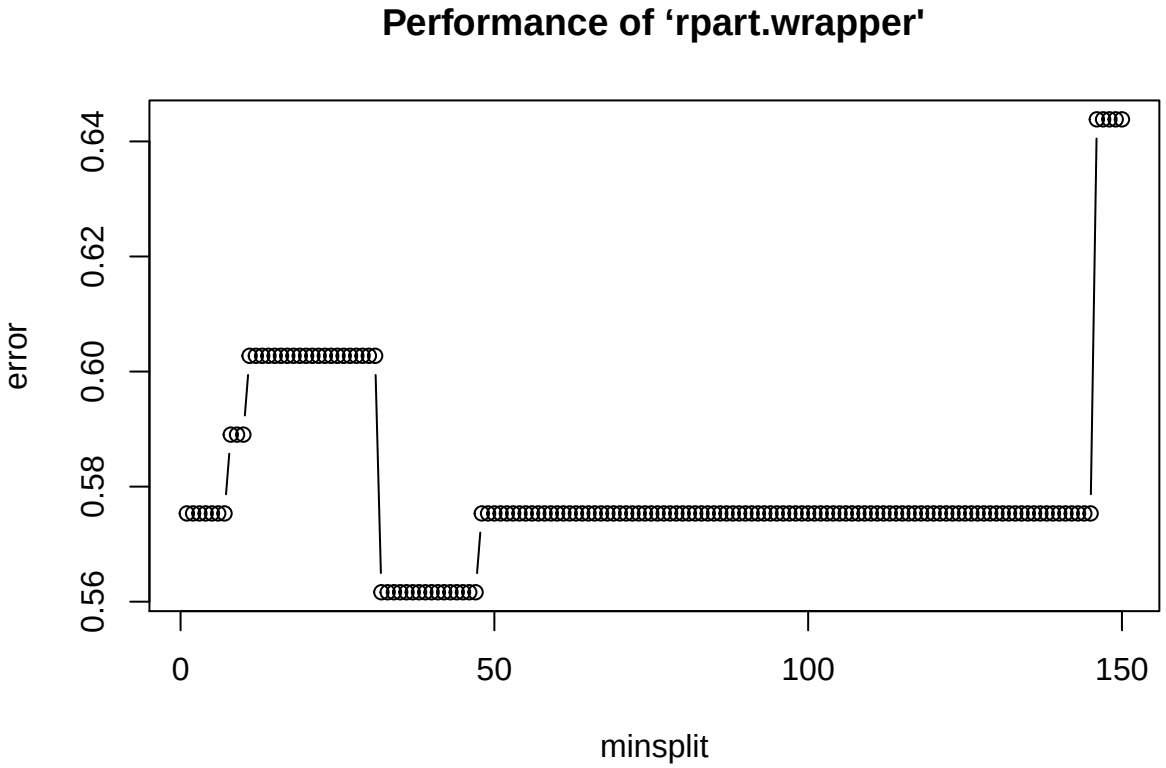
```
##
## Parameter tuning of 'rpart.wrapper':
##
## - sampling method: fixed training/validation set
##
## - best parameters:
##  minsplit
##        32
##
## - best performance: 0.5616438
##
## - Detailed performance results:
##     minsplit     error dispersion
## 1          1 0.5753425         NA
## 2          2 0.5753425         NA
## 3          3 0.5753425         NA
## 4          4 0.5753425         NA
## 5          5 0.5753425         NA
## 6          6 0.5753425         NA
```

```
## 7          7 0.5753425        NA
## 8          8 0.5890411        NA
## 9          9 0.5890411        NA
## 10        10 0.5890411        NA
## 11        11 0.6027397        NA
## 12        12 0.6027397        NA
## 13        13 0.6027397        NA
## 14        14 0.6027397        NA
## 15        15 0.6027397        NA
## 16        16 0.6027397        NA
## 17        17 0.6027397        NA
## 18        18 0.6027397        NA
## 19        19 0.6027397        NA
## 20        20 0.6027397        NA
## 21        21 0.6027397        NA
## 22        22 0.6027397        NA
## 23        23 0.6027397        NA
## 24        24 0.6027397        NA
## 25        25 0.6027397        NA
## 26        26 0.6027397        NA
## 27        27 0.6027397        NA
## 28        28 0.6027397        NA
## 29        29 0.6027397        NA
## 30        30 0.6027397        NA
## 31        31 0.6027397        NA
## 32        32 0.5616438        NA
## 33        33 0.5616438        NA
## 34        34 0.5616438        NA
## 35        35 0.5616438        NA
## 36        36 0.5616438        NA
## 37        37 0.5616438        NA
## 38        38 0.5616438        NA
## 39        39 0.5616438        NA
## 40        40 0.5616438        NA
## 41        41 0.5616438        NA
## 42        42 0.5616438        NA
## 43        43 0.5616438        NA
## 44        44 0.5616438        NA
## 45        45 0.5616438        NA
## 46        46 0.5616438        NA
## 47        47 0.5616438        NA
## 48        48 0.5753425        NA
## 49        49 0.5753425        NA
## 50        50 0.5753425        NA
## 51        51 0.5753425        NA
## 52        52 0.5753425        NA
## 53        53 0.5753425        NA
## 54        54 0.5753425        NA
## 55        55 0.5753425        NA
## 56        56 0.5753425        NA
## 57        57 0.5753425        NA
## 58        58 0.5753425        NA
## 59        59 0.5753425        NA
## 60        60 0.5753425        NA
```

```
## 61        61 0.5753425         NA
## 62        62 0.5753425         NA
## 63        63 0.5753425         NA
## 64        64 0.5753425         NA
## 65        65 0.5753425         NA
## 66        66 0.5753425         NA
## 67        67 0.5753425         NA
## 68        68 0.5753425         NA
## 69        69 0.5753425         NA
## 70        70 0.5753425         NA
## 71        71 0.5753425         NA
## 72        72 0.5753425         NA
## 73        73 0.5753425         NA
## 74        74 0.5753425         NA
## 75        75 0.5753425         NA
## 76        76 0.5753425         NA
## 77        77 0.5753425         NA
## 78        78 0.5753425         NA
## 79        79 0.5753425         NA
## 80        80 0.5753425         NA
## 81        81 0.5753425         NA
## 82        82 0.5753425         NA
## 83        83 0.5753425         NA
## 84        84 0.5753425         NA
## 85        85 0.5753425         NA
## 86        86 0.5753425         NA
## 87        87 0.5753425         NA
## 88        88 0.5753425         NA
## 89        89 0.5753425         NA
## 90        90 0.5753425         NA
## 91        91 0.5753425         NA
## 92        92 0.5753425         NA
## 93        93 0.5753425         NA
## 94        94 0.5753425         NA
## 95        95 0.5753425         NA
## 96        96 0.5753425         NA
## 97        97 0.5753425         NA
## 98        98 0.5753425         NA
## 99        99 0.5753425         NA
## 100      100 0.5753425         NA
## 101      101 0.5753425         NA
## 102      102 0.5753425         NA
## 103      103 0.5753425         NA
## 104      104 0.5753425         NA
## 105      105 0.5753425         NA
## 106      106 0.5753425         NA
## 107      107 0.5753425         NA
## 108      108 0.5753425         NA
## 109      109 0.5753425         NA
## 110      110 0.5753425         NA
## 111      111 0.5753425         NA
## 112      112 0.5753425         NA
## 113      113 0.5753425         NA
## 114      114 0.5753425         NA
```

```
## 115      115 0.5753425         NA
## 116      116 0.5753425         NA
## 117      117 0.5753425         NA
## 118      118 0.5753425         NA
## 119      119 0.5753425         NA
## 120      120 0.5753425         NA
## 121      121 0.5753425         NA
## 122      122 0.5753425         NA
## 123      123 0.5753425         NA
## 124      124 0.5753425         NA
## 125      125 0.5753425         NA
## 126      126 0.5753425         NA
## 127      127 0.5753425         NA
## 128      128 0.5753425         NA
## 129      129 0.5753425         NA
## 130      130 0.5753425         NA
## 131      131 0.5753425         NA
## 132      132 0.5753425         NA
## 133      133 0.5753425         NA
## 134      134 0.5753425         NA
## 135      135 0.5753425         NA
## 136      136 0.5753425         NA
## 137      137 0.5753425         NA
## 138      138 0.5753425         NA
## 139      139 0.5753425         NA
## 140      140 0.5753425         NA
## 141      141 0.5753425         NA
## 142      142 0.5753425         NA
## 143      143 0.5753425         NA
## 144      144 0.5753425         NA
## 145      145 0.5753425         NA
## 146      146 0.6438356         NA
## 147      147 0.6438356         NA
## 148      148 0.6438356         NA
## 149      149 0.6438356         NA
## 150      150 0.6438356         NA
```

```
plot(tuneobj_rpart)
```



Beim Rpart-Modell wird minsplit auf 71 gesetzt

RandomForest:

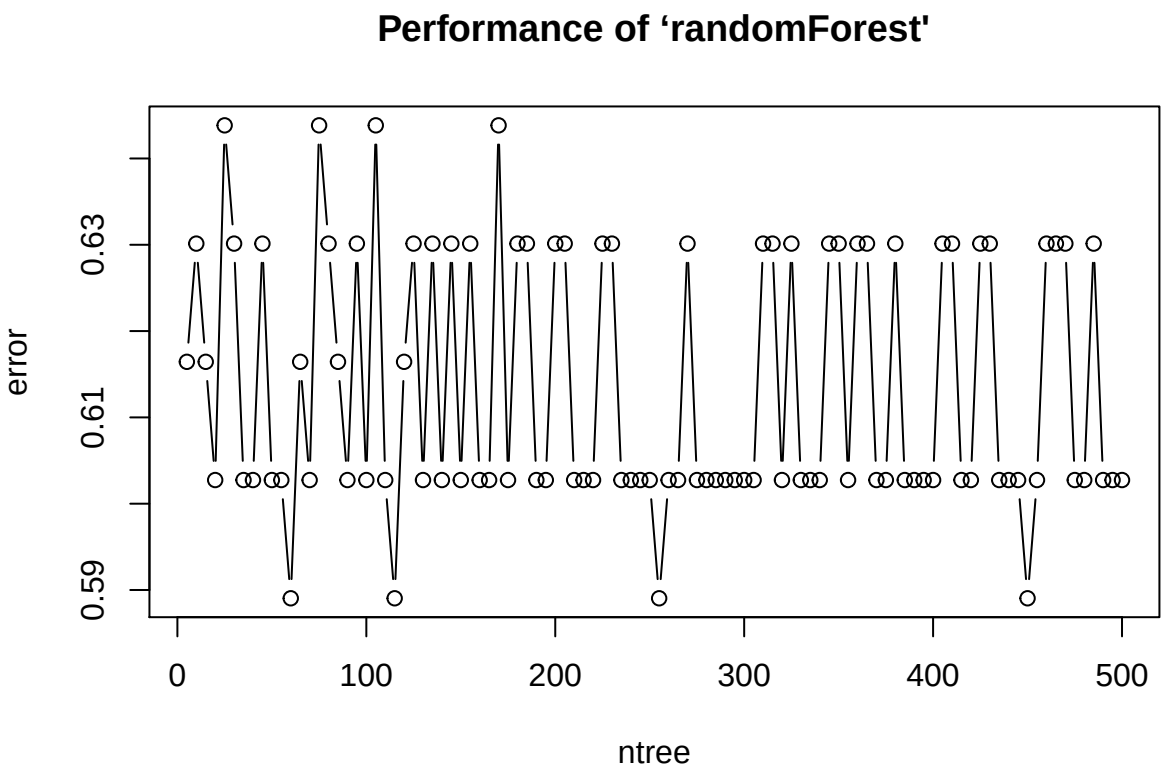
```
tuneobj_rf = tune.randomForest(Rcuisine ~ smoker + drink_level + budget + personality, data = train_cui
                               ntree = 1:100 * 5,
                               tunecontrol = tune.control(sampling = "fix"))

summary(tuneobj_rf)
```

```
##
## Parameter tuning of 'randomForest':
##
## - sampling method: fixed training/validation set
##
## - best parameters:
##  ntree
##     60
##
## - best performance: 0.5890411
##
## - Detailed performance results:
##    ntree     error dispersion
## 1      5 0.6164384         NA
## 2     10 0.6301370         NA
## 3     15 0.6164384         NA
## 4     20 0.6027397         NA
## 5     25 0.6438356         NA
## 6     30 0.6301370         NA
## 7     35 0.6027397         NA
```

```
## 8      40 0.6027397         NA
## 9      45 0.6301370         NA
## 10     50 0.6027397         NA
## 11     55 0.6027397         NA
## 12     60 0.5890411         NA
## 13     65 0.6164384         NA
## 14     70 0.6027397         NA
## 15     75 0.6438356         NA
## 16     80 0.6301370         NA
## 17     85 0.6164384         NA
## 18     90 0.6027397         NA
## 19     95 0.6301370         NA
## 20    100 0.6027397         NA
## 21    105 0.6438356         NA
## 22    110 0.6027397         NA
## 23    115 0.5890411         NA
## 24    120 0.6164384         NA
## 25    125 0.6301370         NA
## 26    130 0.6027397         NA
## 27    135 0.6301370         NA
## 28    140 0.6027397         NA
## 29    145 0.6301370         NA
## 30    150 0.6027397         NA
## 31    155 0.6301370         NA
## 32    160 0.6027397         NA
## 33    165 0.6027397         NA
## 34    170 0.6438356         NA
## 35    175 0.6027397         NA
## 36    180 0.6301370         NA
## 37    185 0.6301370         NA
## 38    190 0.6027397         NA
## 39    195 0.6027397         NA
## 40    200 0.6301370         NA
## 41    205 0.6301370         NA
## 42    210 0.6027397         NA
## 43    215 0.6027397         NA
## 44    220 0.6027397         NA
## 45    225 0.6301370         NA
## 46    230 0.6301370         NA
## 47    235 0.6027397         NA
## 48    240 0.6027397         NA
## 49    245 0.6027397         NA
## 50    250 0.6027397         NA
## 51    255 0.5890411         NA
## 52    260 0.6027397         NA
## 53    265 0.6027397         NA
## 54    270 0.6301370         NA
## 55    275 0.6027397         NA
## 56    280 0.6027397         NA
## 57    285 0.6027397         NA
## 58    290 0.6027397         NA
## 59    295 0.6027397         NA
## 60    300 0.6027397         NA
## 61    305 0.6027397         NA
```

```
## 62    310 0.6301370         NA
## 63    315 0.6301370         NA
## 64    320 0.6027397         NA
## 65    325 0.6301370         NA
## 66    330 0.6027397         NA
## 67    335 0.6027397         NA
## 68    340 0.6027397         NA
## 69    345 0.6301370         NA
## 70    350 0.6301370         NA
## 71    355 0.6027397         NA
## 72    360 0.6301370         NA
## 73    365 0.6301370         NA
## 74    370 0.6027397         NA
## 75    375 0.6027397         NA
## 76    380 0.6301370         NA
## 77    385 0.6027397         NA
## 78    390 0.6027397         NA
## 79    395 0.6027397         NA
## 80    400 0.6027397         NA
## 81    405 0.6301370         NA
## 82    410 0.6301370         NA
## 83    415 0.6027397         NA
## 84    420 0.6027397         NA
## 85    425 0.6301370         NA
## 86    430 0.6301370         NA
## 87    435 0.6027397         NA
## 88    440 0.6027397         NA
## 89    445 0.6027397         NA
## 90    450 0.5890411         NA
## 91    455 0.6027397         NA
## 92    460 0.6301370         NA
## 93    465 0.6301370         NA
## 94    470 0.6301370         NA
## 95    475 0.6027397         NA
## 96    480 0.6027397         NA
## 97    485 0.6301370         NA
## 98    490 0.6027397         NA
## 99    495 0.6027397         NA
## 100   500 0.6027397         NA
```

```
plot(tuneobj_rf)
```



Beim Random-Forest Modell wird ntree bei 135 gewählt.

1.6.3 Fitten der Modelle

1.6.3.1 Rating Modelle

```
model_rpart_rating <- rpart(rating ~ ., data = train_rating_scaled_na_free, minsplit = 47)
model_rf_rating <- randomForest(rating ~ ., data = train_rating_scaled_na_free, ntree = 50)
model_nnet_rating  = nnet(rating ~ ., data = train_rating_scaled_na_free, size = 10, decay = 0.2, linou
```

```
## # weights:  163
## initial  value 776.379271
## iter  10 value 662.824453
## iter  20 value 645.580783
## iter  30 value 635.890221
## iter  40 value 629.526237
## iter  50 value 625.890387
## iter  60 value 624.443192
## iter  70 value 623.626786
## iter  80 value 623.355806
## iter  90 value 623.147771
## iter 100 value 622.932469
## iter 110 value 622.855559
## iter 120 value 622.779445
## iter 130 value 622.545624
## iter 140 value 622.402016
## iter 150 value 622.343914
## iter 160 value 622.269011
## iter 170 value 622.248634
```

```
## iter 180 value 622.247038
## iter 190 value 622.246156
## iter 200 value 622.245576
## final  value 622.245367
## converged
```

```
model_naive_rating <- naiveBayes(rating ~ ., data = train_rating_scaled_na_free)
```

1.6.3.2 Cuisine Modelle

```
model_rpart_cuisine <- rpart(Rcuisine ~ ., data = train_cuisine_scaled_na_free, minsplit = 71)
model_rf_cuisine <- randomForest(Rcuisine ~ ., data = train_cuisine_scaled_na_free, ntree = 135)
model_nnet_cuisine  = nnet(Rcuisine ~ ., data = train_cuisine_scaled_na_free, size = 30, decay = 0.2, li
```

```
## # weights:  487
## initial  value 519.609661
## iter  10 value 367.980993
## iter  20 value 347.277130
## iter  30 value 342.334384
## iter  40 value 340.957691
## iter  50 value 340.193231
## iter  60 value 339.901756
## iter  70 value 339.799043
## iter  80 value 339.738088
## iter  90 value 339.713620
## iter 100 value 339.704834
## iter 110 value 339.699585
## iter 120 value 339.696896
## iter 130 value 339.694924
## iter 140 value 339.693760
## iter 150 value 339.693082
## iter 160 value 339.692774
## iter 170 value 339.692541
## iter 180 value 339.692286
## iter 190 value 339.692048
## iter 200 value 339.691885
## iter 210 value 339.691811
## final  value 339.691788
## converged
```

```
model_naive_cuisine <- naiveBayes(Rcuisine ~ ., data = train_cuisine_scaled_na_free)
```

1.6.4 Predict

1.6.4.1 Predict Rating (Train)

```
fitted_rpart_rating <- predict(model_rpart_rating, train_rating_scaled, type = "class")
fitted_rf_rating <- predict(model_rf_rating, train_rating_scaled, type = "class")
fitted_nnet_rating <- predict(model_nnet_rating, train_rating_scaled, type = "class")
fitted_knn_rating <- knn(train = df[, -1], cl = df$rating, test = df_test[, -1])
fitted_naive_rating <- predict(model_naive_rating, train_rating_scaled)
```

1.6.4.2 Predict Cuisine (Train)

```
fitted_rpart_cuisine <- predict(model_rpart_cuisine, train_cuisine_scaled, type = "class")
fitted_rf_cuisine <- predict(model_rf_cuisine, train_cuisine_scaled, type = "class")
fitted_nnet_cuisine <- predict(model_nnet_cuisine, train_cuisine_scaled, type = "class")
fitted_naive_cuisine <- predict(model_naive_cuisine, train_cuisine_scaled)
```

1.6.4.3 Training Error: Rating Modelle

```
train_error_rpart <- fitted_rpart_rating != train_rating_scaled_na_free$rating
```

```
## Warning in `!=.default`(fitted_rpart_rating,
## train_rating_scaled_na_free$rating): longer object length is not a multiple
## of shorter object length
```

```
## Warning in is.na(e1) | is.na(e2): longer object length is not a multiple of
## shorter object length
```

```
train_error_nnet <- fitted_nnet_rating != train_rating_scaled$rating
train_error_rf <- fitted_rf_rating != train_rating_scaled$rating
train_error_knn <- fitted_knn_rating != train_rating_scaled_na_free$rating
```

```
## Warning in `!=.default`(fitted_knn_rating,
## train_rating_scaled_na_free$rating): longer object length is not a multiple
## of shorter object length
```

```
## Warning in `!=.default`(fitted_knn_rating,
## train_rating_scaled_na_free$rating): longer object length is not a multiple
## of shorter object length
```

```
train_error_naive <- fitted_naive_rating != train_rating_scaled_na_free$rating
```

```
## Warning in `!=.default`(fitted_naive_rating,
## train_rating_scaled_na_free$rating): longer object length is not a multiple
## of shorter object length
```

```
## Warning in `!=.default`(fitted_naive_rating,
## train_rating_scaled_na_free$rating): longer object length is not a multiple
## of shorter object length
```

```
train_error_rf <- na.omit(train_error_rf)
train_error_nnet <- na.omit(train_error_nnet)

c(rating_rpart = mean(train_error_rpart), rating_rf = mean(train_error_rf), rating_knn = mean(train_erro
```

```
## rating_rpart    rating_rf   rating_knn  rating_nnet rating_naive
##    0.6031567    0.3079268    0.6021341    0.4085366    0.5963923
```

1.6.4.4 Training Error: Cuisine Modelle

```
train_error_rpart <- fitted_rpart_cuisine != train_cuisine_scaled_na_free$Rcuisine
```

```
## Warning in `!=.default`(fitted_rpart_cuisine,
## train_cuisine_scaled_na_free$Rcuisine): longer object length is not a
## multiple of shorter object length
```

```
## Warning in is.na(e1) | is.na(e2): longer object length is not a multiple of
## shorter object length
```

```
train_error_nnet <- fitted_nnet_cuisine != train_cuisine_scaled$Rcuisine
train_error_rf <- fitted_rf_cuisine != train_cuisine_scaled$Rcuisine
train_error_naive <- fitted_naive_cuisine != train_cuisine_scaled_na_free$Rcuisine
```

```
## Warning in `!=.default`(fitted_naive_cuisine,
## train_cuisine_scaled_na_free$Rcuisine): longer object length is not a
## multiple of shorter object length
```

```
## Warning in `!=.default`(fitted_naive_cuisine,
## train_cuisine_scaled_na_free$Rcuisine): longer object length is not a
## multiple of shorter object length
```

```
train_error_rf <- na.omit(train_error_rf)
train_error_nnet <- na.omit(train_error_nnet)

c(rating_rpart = mean(train_error_rpart), rating_rf = mean(train_error_rf), rating_nnet = mean(train_er
```

```
## rating_rpart    rating_rf  rating_nnet rating_naive
##    0.7181818    0.5504587    0.5917431    0.7090909
```

1.6.4.5 Predict Rating (Test)

```
fitted_rpart_rating <- predict(model_rpart_rating, test_rating_scaled, type = "class")
fitted_rf_rating <- predict(model_rf_rating, test_rating_scaled, type = "class")
fitted_nnet_rating <- predict(model_nnet_rating, test_rating_scaled_na_free, type = "class")
fitted_knn_rating <- knn(train = df[, -1], cl = df$rating, test = df_test[, -1])
fitted_naive_rating <- predict(model_naive_rating, test_rating_scaled)
```

1.6.4.6 Predict Cuisine (Test)

```
fitted_rpart_cuisine <- predict(model_rpart_cuisine, test_cuisine_scaled, type = "class")
fitted_rf_cuisine <- predict(model_rf_cuisine, test_cuisine_scaled, type = "class")
fitted_nnet_cuisine <- predict(model_nnet_cuisine, test_cuisine_scaled, type = "class")
fitted_naive_cuisine <- predict(model_naive_cuisine, test_cuisine_scaled)
```

1.6.4.7 Generalization error: Rating Modelle

```
test_error_rpart <- fitted_rpart_rating != test_rating_scaled_na_free$rating
```

```
## Warning in `!=.default`(fitted_rpart_rating,
## test_rating_scaled_na_free$rating): longer object length is not a multiple
## of shorter object length
```

```
## Warning in is.na(e1) | is.na(e2): longer object length is not a multiple of
## shorter object length
```

```
test_error_nnet <- fitted_nnet_rating != test_rating_scaled$rating
```

```
## Warning in `!=.default`(fitted_nnet_rating, test_rating_scaled$rating):
## longer object length is not a multiple of shorter object length

## Warning in `!=.default`(fitted_nnet_rating, test_rating_scaled$rating):
## longer object length is not a multiple of shorter object length
```

```
test_error_rf <- fitted_rf_rating != test_rating_scaled$rating
test_error_knn <- fitted_knn_rating != test_rating_scaled_na_free$rating
test_error_naive <- fitted_naive_rating != test_rating_scaled_na_free$rating
```

```
## Warning in `!=.default`(fitted_naive_rating,
## test_rating_scaled_na_free$rating): longer object length is not a multiple
## of shorter object length

## Warning in `!=.default`(fitted_naive_rating,
## test_rating_scaled_na_free$rating): longer object length is not a multiple
## of shorter object length
```

```
test_error_rf <- na.omit(test_error_rf)
test_error_nnet <- na.omit(test_error_nnet)

c(rating_rpart = mean(test_error_rpart), rating_rf = mean(test_error_rf), rating_knn = mean(test_error_
```

```
## rating_rpart    rating_rf   rating_knn  rating_nnet rating_naive
##    0.5765766    0.4953271    0.4922118    0.5765766    0.5698198
```

```
cm_rpart <- confusionMatrix(fitted_rpart_rating, test_rating_scaled$rating)
cm_rf <- confusionMatrix(fitted_rf_rating, test_rating_scaled$rating)
cm_knn <- confusionMatrix(fitted_knn_rating, test_rating_scaled_na_free$rating)
cm_nnet <- confusionMatrix(as.factor(fitted_nnet_rating), test_rating_scaled_na_free$rating)
cm_naive <- confusionMatrix(fitted_naive_rating, test_rating_scaled$rating)
```

RPART:

```
cm_rpart
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1   2
##          0  14   4   3
```

```
##          1  23  55  44
##          2  67 102 132
##
## Overall Statistics
##
##                Accuracy : 0.4527
##                  95% CI : (0.4057, 0.5003)
##     No Information Rate : 0.4032
##     P-Value [Acc > NIR] : 0.01917
##
##                   Kappa : 0.1115
##
##  Mcnemar's Test P-Value : < 2e-16
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2
## Sensitivity           0.13462   0.3416   0.7374
## Specificity           0.97941   0.7633   0.3623
## Pos Pred Value        0.66667   0.4508   0.4385
## Neg Pred Value        0.78723   0.6708   0.6713
## Prevalence            0.23423   0.3626   0.4032
## Detection Rate        0.03153   0.1239   0.2973
## Detection Prevalence  0.04730   0.2748   0.6779
## Balanced Accuracy     0.55701   0.5524   0.5498
```

```
cm_rpart$byClass
```

```
##          Sensitivity Specificity Pos Pred Value Neg Pred Value Precision
## Class: 0   0.1346154   0.9794118      0.6666667      0.7872340 0.6666667
## Class: 1   0.3416149   0.7632509      0.4508197      0.6708075 0.4508197
## Class: 2   0.7374302   0.3622642      0.4385382      0.6713287 0.4385382
##             Recall        F1 Prevalence Detection Rate
## Class: 0 0.1346154 0.2240000  0.2342342     0.03153153
## Class: 1 0.3416149 0.3886926  0.3626126     0.12387387
## Class: 2 0.7374302 0.5500000  0.4031532     0.29729730
##          Detection Prevalence Balanced Accuracy
## Class: 0            0.0472973         0.5570136
## Class: 1            0.2747748         0.5524329
## Class: 2            0.6779279         0.5498472
```

RandomForest:

```
cm_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1  2
##          0 25  6  6
##          1 24 58 45
##          2 23 55 79
##
```

```
## Overall Statistics
##
##                Accuracy : 0.5047
##                  95% CI : (0.4486, 0.5607)
##     No Information Rate : 0.405
##     P-Value [Acc > NIR] : 0.0001921
##
##                   Kappa : 0.213
##
##  Mcnemar's Test P-Value : 7.298e-05
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2
## Sensitivity           0.34722   0.4874   0.6077
## Specificity           0.95181   0.6584   0.5916
## Pos Pred Value        0.67568   0.4567   0.5032
## Neg Pred Value        0.83451   0.6856   0.6890
## Prevalence            0.22430   0.3707   0.4050
## Detection Rate        0.07788   0.1807   0.2461
## Detection Prevalence  0.11526   0.3956   0.4891
## Balanced Accuracy     0.64951   0.5729   0.5997
```

```
cm_rf$byClass
```

```
##          Sensitivity Specificity Pos Pred Value Neg Pred Value Precision
## Class: 0   0.3472222   0.9518072      0.6756757      0.8345070 0.6756757
## Class: 1   0.4873950   0.6584158      0.4566929      0.6855670 0.4566929
## Class: 2   0.6076923   0.5916230      0.5031847      0.6890244 0.5031847
##             Recall        F1 Prevalence Detection Rate
## Class: 0 0.3472222 0.4587156  0.2242991     0.07788162
## Class: 1 0.4873950 0.4715447  0.3707165     0.18068536
## Class: 2 0.6076923 0.5505226  0.4049844     0.24610592
##          Detection Prevalence Balanced Accuracy
## Class: 0            0.1152648         0.6495147
## Class: 1            0.3956386         0.5729054
## Class: 2            0.4890966         0.5996577
```

KNN:

```
cm_knn
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1  2
##          0 24  9 10
##          1 21 59 40
##          2 27 51 80
##
## Overall Statistics
##
##                Accuracy : 0.5078
```

```
##                  95% CI : (0.4517, 0.5637)
##     No Information Rate : 0.405
##     P-Value [Acc > NIR] : 0.0001248
##
##                   Kappa : 0.2212
##
##  Mcnemar's Test P-Value : 0.0029873
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2
## Sensitivity           0.33333   0.4958   0.6154
## Specificity           0.92369   0.6980   0.5916
## Pos Pred Value        0.55814   0.4917   0.5063
## Neg Pred Value        0.82734   0.7015   0.6933
## Prevalence            0.22430   0.3707   0.4050
## Detection Rate        0.07477   0.1838   0.2492
## Detection Prevalence  0.13396   0.3738   0.4922
## Balanced Accuracy     0.62851   0.5969   0.6035
```

```
cm_knn$byClass
```

```
##          Sensitivity Specificity Pos Pred Value Neg Pred Value Precision
## Class: 0   0.3333333   0.9236948      0.5581395      0.8273381 0.5581395
## Class: 1   0.4957983   0.6980198      0.4916667      0.7014925 0.4916667
## Class: 2   0.6153846   0.5916230      0.5063291      0.6932515 0.5063291
##             Recall        F1 Prevalence Detection Rate
## Class: 0 0.3333333 0.4173913  0.2242991     0.07476636
## Class: 1 0.4957983 0.4937238  0.3707165     0.18380062
## Class: 2 0.6153846 0.5555556  0.4049844     0.24922118
##          Detection Prevalence Balanced Accuracy
## Class: 0            0.1339564         0.6285141
## Class: 1            0.3738318         0.5969091
## Class: 2            0.4922118         0.6035038
```

NNET:

```
cm_nnet
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1  2
##          0 12  5  4
##          1 28 57 43
##          2 32 57 83
##
## Overall Statistics
##
##                Accuracy : 0.4735
##                  95% CI : (0.4178, 0.5297)
##     No Information Rate : 0.405
##     P-Value [Acc > NIR] : 0.007548
```

```
##
##                   Kappa : 0.1515
##
##  Mcnemar's Test P-Value : 1.193e-08
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2
## Sensitivity           0.16667   0.4790   0.6385
## Specificity           0.96386   0.6485   0.5340
## Pos Pred Value        0.57143   0.4453   0.4826
## Neg Pred Value        0.80000   0.6788   0.6846
## Prevalence            0.22430   0.3707   0.4050
## Detection Rate        0.03738   0.1776   0.2586
## Detection Prevalence  0.06542   0.3988   0.5358
## Balanced Accuracy     0.56526   0.5638   0.5862
```

```
cm_nnet$byClass
```

```
##          Sensitivity Specificity Pos Pred Value Neg Pred Value Precision
## Class: 0   0.1666667   0.9638554      0.5714286      0.8000000 0.5714286
## Class: 1   0.4789916   0.6485149      0.4453125      0.6787565 0.4453125
## Class: 2   0.6384615   0.5340314      0.4825581      0.6845638 0.4825581
##             Recall        F1 Prevalence Detection Rate
## Class: 0 0.1666667 0.2580645  0.2242991     0.03738318
## Class: 1 0.4789916 0.4615385  0.3707165     0.17757009
## Class: 2 0.6384615 0.5496689  0.4049844     0.25856698
##          Detection Prevalence Balanced Accuracy
## Class: 0           0.06542056         0.5652610
## Class: 1           0.39875389         0.5637532
## Class: 2           0.53582555         0.5862465
```

Naive Bayes:

```
cm_naive
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1  2
##          0  0  0  0
##          1 68 96 97
##          2 36 65 82
##
## Overall Statistics
##
##                Accuracy : 0.4009
##                  95% CI : (0.355, 0.4481)
##     No Information Rate : 0.4032
##     P-Value [Acc > NIR] : 0.5565
##
##                   Kappa : 0.0348
##
```

```
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2
## Sensitivity            0.0000   0.5963   0.4581
## Specificity            1.0000   0.4170   0.6189
## Pos Pred Value            NaN   0.3678   0.4481
## Neg Pred Value         0.7658   0.6448   0.6284
## Prevalence             0.2342   0.3626   0.4032
## Detection Rate         0.0000   0.2162   0.1847
## Detection Prevalence   0.0000   0.5878   0.4122
## Balanced Accuracy      0.5000   0.5066   0.5385
```

```
cm_naive$byClass
```

```
##          Sensitivity Specificity Pos Pred Value Neg Pred Value Precision
## Class: 0   0.0000000   1.0000000            NaN      0.7657658        NA
## Class: 1   0.5962733   0.4169611      0.3678161      0.6448087 0.3678161
## Class: 2   0.4581006   0.6188679      0.4480874      0.6283525 0.4480874
##             Recall        F1 Prevalence Detection Rate
## Class: 0 0.0000000        NA  0.2342342      0.0000000
## Class: 1 0.5962733 0.4549763  0.3626126      0.2162162
## Class: 2 0.4581006 0.4530387  0.4031532      0.1846847
##          Detection Prevalence Balanced Accuracy
## Class: 0            0.0000000         0.5000000
## Class: 1            0.5878378         0.5066172
## Class: 2            0.4121622         0.5384842
```

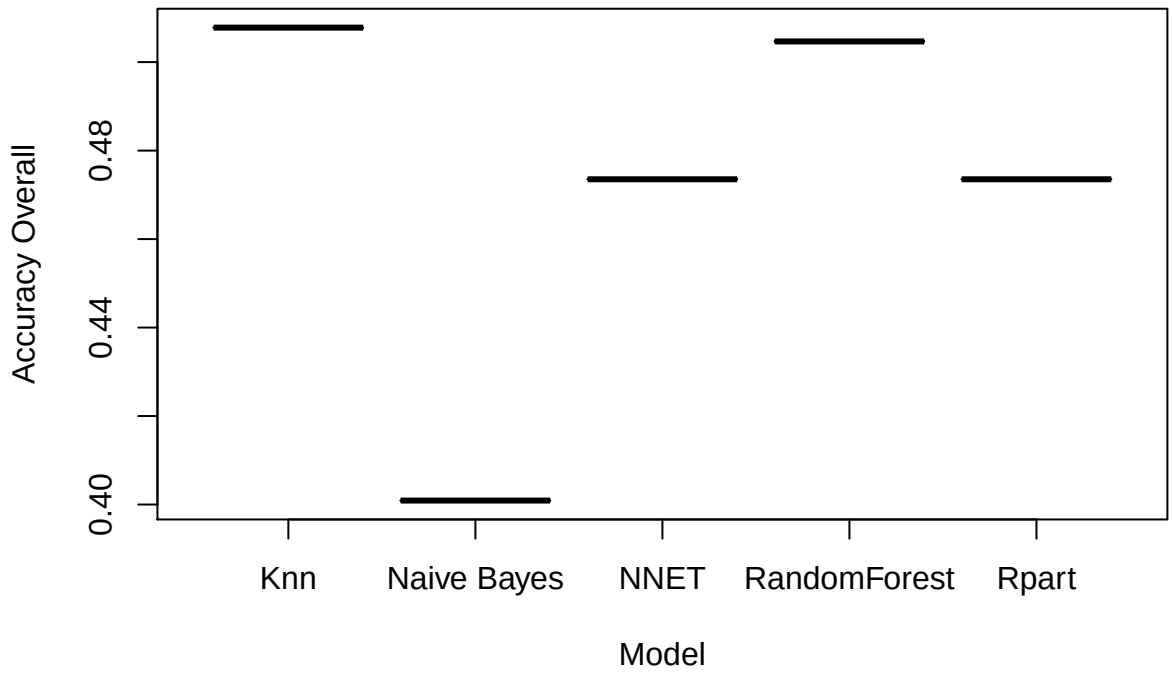
1.6.4.8 Conclusion - Rating

```
colnames <- c("Naive Bayes", "Knn", "RandomForest", "Rpart", "NNET")

spec_1 <- c(cm_naive$byClass[1, "Specificity"], cm_knn$byClass[1, "Specificity"], cm_rf$byClass[1, "Spec
spec_2 <- c(cm_naive$byClass[2, "Specificity"], cm_knn$byClass[2, "Specificity"], cm_rf$byClass[2, "Spec
spec_3 <- c(cm_naive$byClass[3, "Specificity"], cm_knn$byClass[3, "Specificity"], cm_rf$byClass[3, "Spec

prec_1 <- c(cm_naive$byClass[1, "Precision"], cm_knn$byClass[1, "Precision"], cm_rf$byClass[1, "Precisi
prec_2 <- c(cm_naive$byClass[2, "Precision"], cm_knn$byClass[2, "Precision"], cm_rf$byClass[2, "Precisi
prec_3 <- c(cm_naive$byClass[3, "Precision"], cm_knn$byClass[3, "Precision"], cm_rf$byClass[3, "Precisi

rec_1 <- c(cm_naive$byClass[1, "Recall"], cm_knn$byClass[1, "Recall"], cm_rf$byClass[1, "Recall"], cm_rp
rec_2 <- c(cm_naive$byClass[2, "Recall"], cm_knn$byClass[2, "Recall"], cm_rf$byClass[2, "Recall"], cm_rp
rec_3 <- c(cm_naive$byClass[3, "Recall"], cm_knn$byClass[3, "Recall"], cm_rf$byClass[3, "Recall"], cm_rp

f1_1 <- c(cm_naive$byClass[1, "F1"], cm_knn$byClass[1, "F1"], cm_rf$byClass[1, "F1"], cm_rpart$byClass[
f1_2 <- c(cm_naive$byClass[2, "F1"], cm_knn$byClass[2, "F1"], cm_rf$byClass[2, "F1"], cm_rpart$byClass[
f1_3 <- c(cm_naive$byClass[3, "F1"], cm_knn$byClass[3, "F1"], cm_rf$byClass[3, "F1"], cm_rpart$byClass[

acc_1 <- c(cm_naive$byClass[1, "Balanced Accuracy"], cm_knn$byClass[1, "Balanced Accuracy"],
           cm_rf$byClass[1, "Balanced Accuracy"], cm_rpart$byClass[1, "Balanced Accuracy"], cm_nnet$byCl
acc_2 <- c(cm_naive$byClass[2, "Balanced Accuracy"], cm_knn$byClass[2, "Balanced Accuracy"],
```

```
          cm_rf$byClass[2, "Balanced Accuracy"], cm_rpart$byClass[2, "Balanced Accuracy"], cm_nnet$byCl
acc_3 <- c(cm_naive$byClass[3, "Balanced Accuracy"], cm_knn$byClass[3, "Balanced Accuracy"],
          cm_rf$byClass[3, "Balanced Accuracy"], cm_rpart$byClass[3, "Balanced Accuracy"], cm_nnet$byCl

overall_acc <- c(cm_naive$overall["Accuracy"], cm_knn$overall["Accuracy"], cm_rf$overall["Accuracy"], cm

comp_all = data.frame(colnames, spec_1, spec_2, spec_3, prec_1, prec_2, prec_3, f1_1, f1_2, f1_3, acc_1

comp_spec = data.frame(colnames, spec_1, spec_2, spec_3)
colnames(comp_spec) <- c("Model", "Specificity Class-1", "Specificity Class-2", "Specificity Class-3")

comp_prec = data.frame(colnames, prec_1, prec_2, prec_3)
colnames(comp_prec) <- c("Model", "Precision Class-1", "Precision Class-2", "Precision Class-3")

comp_rec = data.frame(colnames, rec_1, rec_2, rec_3)
colnames(comp_rec) <- c("Model", "Recall Class-1", "Recall Class-2", "Recall Class-3")

comp_f1 = data.frame(colnames, f1_1, f1_2, f1_3)
colnames(comp_f1) <- c("Model", "F1 Class-1", "F1 Class-2", "F1 Class-3")

comp_acc = data.frame(colnames, acc_1, acc_2, acc_3)
colnames(comp_acc) <- c("Model", "Accuracy Class-1", "Accuracy Class-2", "Accuracy Class-3")

comp_all_acc = data.frame(colnames, overall_acc)
colnames(comp_all_acc) <- c("Model", "Accuracy Overall")
```

Overall Accuracy:

```
plot(comp_all_acc)
```



Im Diagramm ist deutlich zu sehen, dass KNN und Random Forest die höchste Overall Accuracy haben.

Specification:

```
comp_spec
```

```
##          Model Specificity Class-1 Specificity Class-2 Specificity Class-3
## 1  Naive Bayes           1.0000000           0.4169611           0.6188679
## 2          Knn           0.9236948           0.6980198           0.5916230
## 3 RandomForest           0.9518072           0.6584158           0.5916230
## 4        Rpart           0.9794118           0.7632509           0.3622642
## 5         NNET           0.9638554           0.6485149           0.5340314
```

Über alle 3 Klassen gesehen ist ersichtlich, dass NNET, KNN und Random Forest eine fasst gleich gute Specificity haben. Von diesen Top 3 hat das NNET Modell die schlechteste Specificity.

Recall:

```
comp_rec
```

```
##          Model Recall Class-1 Recall Class-2 Recall Class-3
## 1  Naive Bayes      0.0000000      0.5962733      0.4581006
## 2          Knn      0.3333333      0.4957983      0.6153846
## 3 RandomForest      0.3472222      0.4873950      0.6076923
## 4        Rpart      0.1346154      0.3416149      0.7374302
## 5         NNET      0.1666667      0.4789916      0.6384615
```

Ähnlich wie bei der Specification haben auch hier KNN und Random Forest annäherend gleiche Werte. Vergleicht man diese mit den anderen Methoden haben diese zwei die besten Werte.

Precision:

```
comp_prec
```

```
##          Model Precision Class-1 Precision Class-2 Precision Class-3
## 1  Naive Bayes                NA         0.3678161         0.4480874
## 2          Knn         0.5581395         0.4916667         0.5063291
## 3 RandomForest         0.6756757         0.4566929         0.5031847
## 4        Rpart         0.6666667         0.4508197         0.4385382
## 5         NNET         0.5714286         0.4453125         0.4825581
```

Wie auch bei den Werten davor haben auch hier KNN und Random Forest die beste Precision. RPart kann da auch mithalten. Aufgrund des deutlich höheren Precision in der Class 1, die die anderen Modelle n den anderen Klassen nicht ausgleicht, hat Random Forest die beste Precision.

F1-Value:

```
comp_f1
```

```
##          Model F1 Class-1 F1 Class-2 F1 Class-3
## 1  Naive Bayes         NA  0.4549763  0.4530387
## 2          Knn  0.4173913  0.4937238  0.5555556
## 3 RandomForest  0.4587156  0.4715447  0.5505226
## 4        Rpart  0.2240000  0.3886926  0.5500000
## 5         NNET  0.2580645  0.4615385  0.5496689
```

Über die Klassen hinweg hat auch hier Random Forest im Vergleich den besten F1-Value.

1.6.4.9 Generalization error: Cuisine Modelle

```
test_error_rpart <- fitted_rpart_cuisine != test_cuisine_scaled_na_free$Rcuisine
```

```
## Warning in `!=.default`(fitted_rpart_cuisine,
## test_cuisine_scaled_na_free$Rcuisine): longer object length is not a
## multiple of shorter object length
```

```
## Warning in is.na(e1) | is.na(e2): longer object length is not a multiple of
## shorter object length
```

```
test_error_nnet <- fitted_nnet_cuisine != test_cuisine_scaled$Rcuisine
test_error_rf <- fitted_rf_cuisine != test_cuisine_scaled$Rcuisine
test_error_naive <- fitted_naive_cuisine != test_cuisine_scaled_na_free$Rcuisine
```

```
## Warning in `!=.default`(fitted_naive_cuisine,
## test_cuisine_scaled_na_free$Rcuisine): longer object length is not a
## multiple of shorter object length
```

```
## Warning in `!=.default`(fitted_naive_cuisine,
## test_cuisine_scaled_na_free$Rcuisine): longer object length is not a
## multiple of shorter object length
```

```
test_error_rf <- na.omit(test_error_rf)
test_error_nnet <- na.omit(test_error_nnet)

c(cuisine_rpart = mean(test_error_rpart), cuisine_rf = mean(test_error_rf), cuisine_nnet = mean(test_er
```

```
## cuisine_rpart    cuisine_rf  cuisine_nnet cuisine_naive
##     0.7090909     0.6476190     0.6476190     0.6636364
```

```
cm_rpart <- confusionMatrix(fitted_rpart_cuisine, test_cuisine_scaled$Rcuisine)
cm_rf <- confusionMatrix(fitted_rf_cuisine, test_cuisine_scaled$Rcuisine)
cm_nnet <- confusionMatrix(as.factor(fitted_nnet_cuisine), test_cuisine_scaled$Rcuisine)
```

```
## Warning in levels(reference) != levels(data): longer object length is not a
## multiple of shorter object length
```

```
## Warning in confusionMatrix.default(as.factor(fitted_nnet_cuisine),
## test_cuisine_scaled$Rcuisine): Levels are not in the same order for
## reference and data. Refactoring data to match.
```

```
cm_naive <- confusionMatrix(fitted_naive_cuisine, test_cuisine_scaled$Rcuisine)
```

RPART:

```
cm_rpart
```

```
## Confusion Matrix and Statistics
##
##                 Reference
## Prediction       Persian African American Asian International European
##   Persian              0       0        0     0             0        0
##   African              0       0        0     0             0        0
##   American             1       0        2     3             1        1
##   Asian                0       0        0     0             0        0
##   International        0       1        7     7             6        8
##   European             0       0        0     0             0        0
##   South_American       2       1        8     5            11        5
##                 Reference
## Prediction       South_American
##   Persian                     0
##   African                     0
##   American                    6
##   Asian                       0
##   International               6
##   European                    0
##   South_American             29
##
## Overall Statistics
##
##                Accuracy : 0.3364
##                  95% CI : (0.2491, 0.4327)
##     No Information Rate : 0.3727
##     P-Value [Acc > NIR] : 0.812
##
##                   Kappa : 0.0803
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Persian Class: African Class: American
## Sensitivity                 0.00000        0.00000         0.11765
## Specificity                 1.00000        1.00000         0.87097
## Pos Pred Value                  NaN            NaN         0.14286
## Neg Pred Value              0.97273        0.98182         0.84375
## Prevalence                  0.02727        0.01818         0.15455
## Detection Rate              0.00000        0.00000         0.01818
## Detection Prevalence        0.00000        0.00000         0.12727
## Balanced Accuracy           0.50000        0.50000         0.49431
##                      Class: Asian Class: International Class: European
## Sensitivity                0.0000              0.33333          0.0000
## Specificity                1.0000              0.68478          1.0000
## Pos Pred Value                NaN              0.17143             NaN
## Neg Pred Value             0.8636              0.84000          0.8727
## Prevalence                 0.1364              0.16364          0.1273
## Detection Rate             0.0000              0.05455          0.0000
## Detection Prevalence       0.0000              0.31818          0.0000
## Balanced Accuracy          0.5000              0.50906          0.5000
##                      Class: South_American
## Sensitivity                         0.7073
```

```
## Specificity                          0.5362
## Pos Pred Value                       0.4754
## Neg Pred Value                       0.7551
## Prevalence                           0.3727
## Detection Rate                       0.2636
## Detection Prevalence                 0.5545
## Balanced Accuracy                    0.6218
```

```
cm_rpart$byClass
```

```
##                      Sensitivity Specificity Pos Pred Value
## Class: Persian         0.0000000   1.0000000            NaN
## Class: African         0.0000000   1.0000000            NaN
## Class: American        0.1176471   0.8709677      0.1428571
## Class: Asian           0.0000000   1.0000000            NaN
## Class: International   0.3333333   0.6847826      0.1714286
## Class: European        0.0000000   1.0000000            NaN
## Class: South_American  0.7073171   0.5362319      0.4754098
##                      Neg Pred Value Precision    Recall        F1
## Class: Persian            0.9727273        NA 0.0000000        NA
## Class: African            0.9818182        NA 0.0000000        NA
## Class: American           0.8437500 0.1428571 0.1176471 0.1290323
## Class: Asian              0.8636364        NA 0.0000000        NA
## Class: International      0.8400000 0.1714286 0.3333333 0.2264151
## Class: European           0.8727273        NA 0.0000000        NA
## Class: South_American     0.7551020 0.4754098 0.7073171 0.5686275
##                      Prevalence Detection Rate Detection Prevalence
## Class: Persian       0.02727273     0.00000000            0.0000000
## Class: African       0.01818182     0.00000000            0.0000000
## Class: American      0.15454545     0.01818182            0.1272727
## Class: Asian         0.13636364     0.00000000            0.0000000
## Class: International 0.16363636     0.05454545            0.3181818
## Class: European      0.12727273     0.00000000            0.0000000
## Class: South_American 0.37272727     0.26363636            0.5545455
##                      Balanced Accuracy
## Class: Persian               0.5000000
## Class: African               0.5000000
## Class: American              0.4943074
## Class: Asian                 0.5000000
## Class: International         0.5090580
## Class: European              0.5000000
## Class: South_American        0.6217745
```

RandomForest:

```
cm_rf
```

```
## Confusion Matrix and Statistics
##
##                 Reference
## Prediction       Persian African American Asian International European
##   Persian              0       0        0     0             0        0
##   African              0       0        0     0             0        0
```

```
##   American             1       0        2     3             1        1
##   Asian                0       0        0     0             0        0
##   International        1       2       10     8             8       11
##   European             0       0        0     0             0        0
##   South_American       0       0        5     4             9        2
##                Reference
## Prediction       South_American
##   Persian                     0
##   African                     0
##   American                    3
##   Asian                       0
##   International               7
##   European                    0
##   South_American             27
##
## Overall Statistics
##
##                Accuracy : 0.3524
##                  95% CI : (0.2616, 0.4517)
##     No Information Rate : 0.3524
##     P-Value [Acc > NIR] : 0.5367
##
##                   Kappa : 0.1349
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Persian Class: African Class: American
## Sensitivity                 0.00000        0.00000         0.11765
## Specificity                 1.00000        1.00000         0.89773
## Pos Pred Value                  NaN            NaN         0.18182
## Neg Pred Value              0.98095        0.98095         0.84043
## Prevalence                  0.01905        0.01905         0.16190
## Detection Rate              0.00000        0.00000         0.01905
## Detection Prevalence        0.00000        0.00000         0.10476
## Balanced Accuracy           0.50000        0.50000         0.50769
##                      Class: Asian Class: International Class: European
## Sensitivity                0.0000              0.44444          0.0000
## Specificity                1.0000              0.55172          1.0000
## Pos Pred Value                NaN              0.17021             NaN
## Neg Pred Value             0.8571              0.82759          0.8667
## Prevalence                 0.1429              0.17143          0.1333
## Detection Rate             0.0000              0.07619          0.0000
## Detection Prevalence       0.0000              0.44762          0.0000
## Balanced Accuracy          0.5000              0.49808          0.5000
##                      Class: South_American
## Sensitivity                         0.7297
## Specificity                         0.7059
## Pos Pred Value                      0.5745
## Neg Pred Value                      0.8276
## Prevalence                          0.3524
## Detection Rate                      0.2571
## Detection Prevalence                0.4476
```

```
## Balanced Accuracy                  0.7178
```

```
cm_rf$byClass
```

```
##                       Sensitivity Specificity Pos Pred Value
## Class: Persian          0.0000000   1.0000000            NaN
## Class: African          0.0000000   1.0000000            NaN
## Class: American         0.1176471   0.8977273      0.1818182
## Class: Asian            0.0000000   1.0000000            NaN
## Class: International    0.4444444   0.5517241      0.1702128
## Class: European         0.0000000   1.0000000            NaN
## Class: South_American   0.7297297   0.7058824      0.5744681
##                       Neg Pred Value Precision    Recall        F1
## Class: Persian             0.9809524        NA 0.0000000        NA
## Class: African             0.9809524        NA 0.0000000        NA
## Class: American            0.8404255 0.1818182 0.1176471 0.1428571
## Class: Asian               0.8571429        NA 0.0000000        NA
## Class: International       0.8275862 0.1702128 0.4444444 0.2461538
## Class: European            0.8666667        NA 0.0000000        NA
## Class: South_American      0.8275862 0.5744681 0.7297297 0.6428571
##                       Prevalence Detection Rate Detection Prevalence
## Class: Persian        0.01904762     0.00000000            0.0000000
## Class: African        0.01904762     0.00000000            0.0000000
## Class: American       0.16190476     0.01904762            0.1047619
## Class: Asian          0.14285714     0.00000000            0.0000000
## Class: International  0.17142857     0.07619048            0.4476190
## Class: European       0.13333333     0.00000000            0.0000000
## Class: South_American 0.35238095     0.25714286            0.4476190
##                       Balanced Accuracy
## Class: Persian                0.5000000
## Class: African                0.5000000
## Class: American               0.5076872
## Class: Asian                  0.5000000
## Class: International          0.4980843
## Class: European               0.5000000
## Class: South_American         0.7178060
```

NNET:

```
cm_nnet
```

```
## Confusion Matrix and Statistics
##
##                  Reference
## Prediction        Persian African American Asian International European
##   Persian               0       0        0     0             0        0
##   African               0       0        0     0             0        0
##   American              0       0        0     0             1        0
##   Asian                 0       0        0     0             0        0
##   International         1       2       10     8             8       12
##   European              0       0        0     0             0        0
##   South_American        1       0        7     7             9        2
##                  Reference
```

```
## Prediction       South_American
##   Persian                     0
##   African                     0
##   American                    1
##   Asian                       0
##   International               7
##   European                    0
##   South_American             29
##
## Overall Statistics
##
##                Accuracy : 0.3524
##                  95% CI : (0.2616, 0.4517)
##     No Information Rate : 0.3524
##     P-Value [Acc > NIR] : 0.5367
##
##                   Kappa : 0.1176
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Persian Class: African Class: American
## Sensitivity                 0.00000        0.00000         0.00000
## Specificity                 1.00000        1.00000         0.97727
## Pos Pred Value                  NaN            NaN         0.00000
## Neg Pred Value              0.98095        0.98095         0.83495
## Prevalence                  0.01905        0.01905         0.16190
## Detection Rate              0.00000        0.00000         0.00000
## Detection Prevalence        0.00000        0.00000         0.01905
## Balanced Accuracy           0.50000        0.50000         0.48864
##                      Class: Asian Class: International Class: European
## Sensitivity                0.0000              0.44444          0.0000
## Specificity                1.0000              0.54023          1.0000
## Pos Pred Value                NaN              0.16667             NaN
## Neg Pred Value             0.8571              0.82456          0.8667
## Prevalence                 0.1429              0.17143          0.1333
## Detection Rate             0.0000              0.07619          0.0000
## Detection Prevalence       0.0000              0.45714          0.0000
## Balanced Accuracy          0.5000              0.49234          0.5000
##                      Class: South_American
## Sensitivity                         0.7838
## Specificity                         0.6176
## Pos Pred Value                      0.5273
## Neg Pred Value                      0.8400
## Prevalence                          0.3524
## Detection Rate                      0.2762
## Detection Prevalence                0.5238
## Balanced Accuracy                   0.7007
```

```
cm_nnet$byClass
```

```
##                      Sensitivity Specificity Pos Pred Value
## Class: Persian         0.0000000   1.0000000            NaN
```

```
## Class: African          0.0000000   1.0000000            NaN
## Class: American         0.0000000   0.9772727      0.0000000
## Class: Asian            0.0000000   1.0000000            NaN
## Class: International    0.4444444   0.5402299      0.1666667
## Class: European         0.0000000   1.0000000            NaN
## Class: South_American   0.7837838   0.6176471      0.5272727
##                       Neg Pred Value Precision    Recall        F1
## Class: Persian             0.9809524        NA 0.0000000        NA
## Class: African             0.9809524        NA 0.0000000        NA
## Class: American            0.8349515 0.0000000 0.0000000       NaN
## Class: Asian               0.8571429        NA 0.0000000        NA
## Class: International       0.8245614 0.1666667 0.4444444 0.2424242
## Class: European            0.8666667        NA 0.0000000        NA
## Class: South_American      0.8400000 0.5272727 0.7837838 0.6304348
##                       Prevalence Detection Rate Detection Prevalence
## Class: Persian        0.01904762     0.00000000           0.00000000
## Class: African        0.01904762     0.00000000           0.00000000
## Class: American       0.16190476     0.00000000           0.01904762
## Class: Asian          0.14285714     0.00000000           0.00000000
## Class: International  0.17142857     0.07619048           0.45714286
## Class: European       0.13333333     0.00000000           0.00000000
## Class: South_American 0.35238095     0.27619048           0.52380952
##                       Balanced Accuracy
## Class: Persian                0.5000000
## Class: African                0.5000000
## Class: American               0.4886364
## Class: Asian                  0.5000000
## Class: International          0.4923372
## Class: European               0.5000000
## Class: South_American         0.7007154
```

Naive Bayes:

```
cm_naive
```

```
## Confusion Matrix and Statistics
##
##                 Reference
## Prediction       Persian African American Asian International European
##   Persian              0       0        0     0             0        0
##   African              0       0        0     0             0        0
##   American             0       0        0     0             1        0
##   Asian                0       0        0     0             0        0
##   International        0       1        7     7             6        8
##   European             0       0        0     0             0        0
##   South_American       3       1       10     8            11        6
##                 Reference
## Prediction       South_American
##   Persian                     0
##   African                     0
##   American                    0
##   Asian                       1
##   International               5
```

```
##   European                   0
##   South_American            35
##
## Overall Statistics
##
##                Accuracy : 0.3727
##                  95% CI : (0.2824, 0.4701)
##     No Information Rate : 0.3727
##     P-Value [Acc > NIR] : 0.5359
##
##                   Kappa : 0.0988
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Persian Class: African Class: American
## Sensitivity                 0.00000        0.00000        0.000000
## Specificity                 1.00000        1.00000        0.989247
## Pos Pred Value                  NaN            NaN        0.000000
## Neg Pred Value              0.97273        0.98182        0.844037
## Prevalence                  0.02727        0.01818        0.154545
## Detection Rate              0.00000        0.00000        0.000000
## Detection Prevalence        0.00000        0.00000        0.009091
## Balanced Accuracy           0.50000        0.50000        0.494624
##                      Class: Asian Class: International Class: European
## Sensitivity              0.000000              0.33333          0.0000
## Specificity              0.989474              0.69565          1.0000
## Pos Pred Value           0.000000              0.17647             NaN
## Neg Pred Value           0.862385              0.84211          0.8727
## Prevalence               0.136364              0.16364          0.1273
## Detection Rate           0.000000              0.05455          0.0000
## Detection Prevalence     0.009091              0.30909          0.0000
## Balanced Accuracy        0.494737              0.51449          0.5000
##                      Class: South_American
## Sensitivity                         0.8537
## Specificity                         0.4348
## Pos Pred Value                      0.4730
## Neg Pred Value                      0.8333
## Prevalence                          0.3727
## Detection Rate                      0.3182
## Detection Prevalence                0.6727
## Balanced Accuracy                   0.6442
```

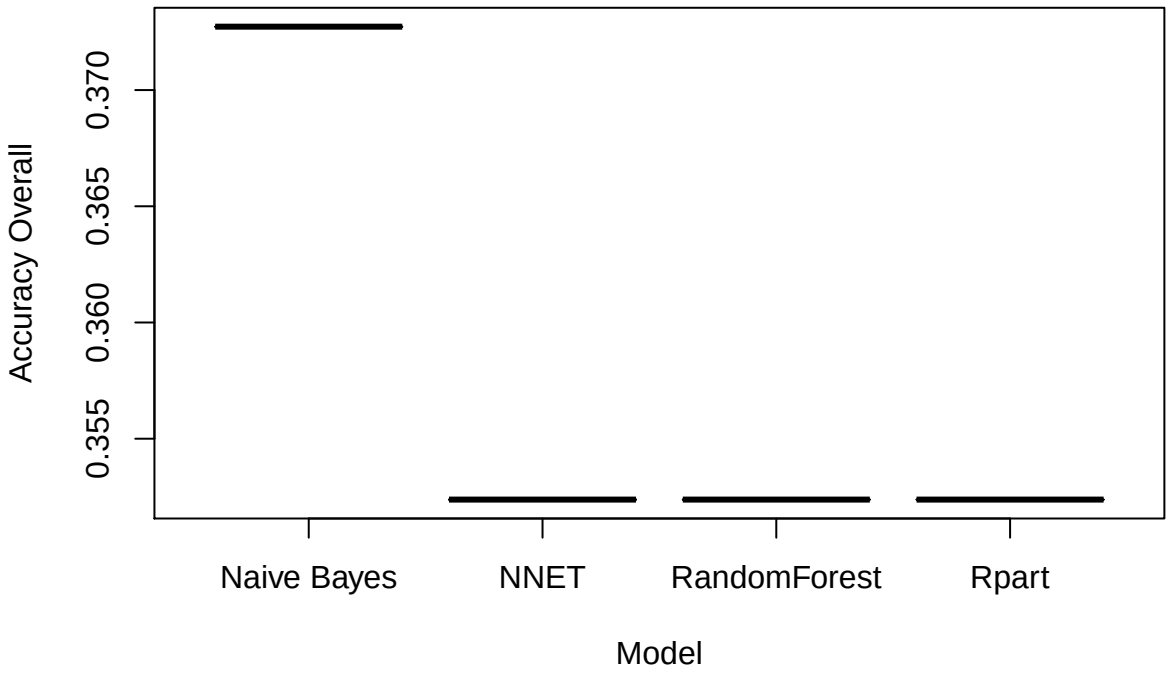
```
cm_naive$byClass
```

```
##                       Sensitivity Specificity Pos Pred Value
## Class: Persian          0.0000000   1.0000000            NaN
## Class: African          0.0000000   1.0000000            NaN
## Class: American         0.0000000   0.9892473      0.0000000
## Class: Asian            0.0000000   0.9894737      0.0000000
## Class: International    0.3333333   0.6956522      0.1764706
## Class: European         0.0000000   1.0000000            NaN
## Class: South_American   0.8536585   0.4347826      0.4729730
```

```
##                       Neg Pred Value Precision    Recall        F1
## Class: Persian             0.9727273        NA 0.0000000        NA
## Class: African             0.9818182        NA 0.0000000        NA
## Class: American            0.8440367 0.0000000 0.0000000       NaN
## Class: Asian               0.8623853 0.0000000 0.0000000       NaN
## Class: International       0.8421053 0.1764706 0.3333333 0.2307692
## Class: European            0.8727273        NA 0.0000000        NA
## Class: South_American      0.8333333 0.4729730 0.8536585 0.6086957
##                       Prevalence Detection Rate Detection Prevalence
## Class: Persian        0.02727273     0.00000000          0.000000000
## Class: African        0.01818182     0.00000000          0.000000000
## Class: American       0.15454545     0.00000000          0.009090909
## Class: Asian          0.13636364     0.00000000          0.009090909
## Class: International  0.16363636     0.05454545          0.309090909
## Class: European       0.12727273     0.00000000          0.000000000
## Class: South_American 0.37272727     0.31818182          0.672727273
##                       Balanced Accuracy
## Class: Persian                0.5000000
## Class: African                0.5000000
## Class: American               0.4946237
## Class: Asian                  0.4947368
## Class: International          0.5144928
## Class: European               0.5000000
## Class: South_American         0.6442206
```

1.6.4.10 Conclusion - Cuisine

```
colnames <- c("Naive Bayes", "RandomForest", "Rpart", "NNET")

overall_acc <- c(cm_naive$overall["Accuracy"], cm_rf$overall["Accuracy"], cm_nnet$overall["Accuracy"], c

comp_all_acc = data.frame(colnames, overall_acc)
colnames(comp_all_acc) <- c("Model", "Accuracy Overall")

plot(comp_all_acc)
```



1.6.5 Conclusion

Für die Vorhersage des Ratings ist das Random Forest Model zu verwenden.

Für das zweite Vorhersagemodell (Cuisine) ist das Naive Bayes Model zu wählen.

1.7 Deployment des besten Modells mittels Web Service [10%]

Über beide Vorhersagen hinweg gesehen, ist das genaueste Model: Random Forest für die Rating Vorgersage. Dieses wird somit auch deployed.

siehe: * model.R * server.R

1.8 Kurzpräsentation des Projekts/der Ergebnisse mittels Dashboard [10%]

siehe app.R

1.9 Extra-Feature - zB neue Methoden, interaktive Visualisierung [20%]

siehe app.R