Projektarbeit

Laurenz Hinterholzer & Manuel Karlsberger

07/31/2020

Aufgabe 1

Tabelle 1 in Aiello et al. (2020) gibt einen Überblick über die Regionen je nach Aggregationsgrad nach Anzahl, durchschnittlicher Fläche und durchschnittlicher Anzahl an Bevölkerung. Die Daten dazu sind enthalten in den Dateien year_*_grocery.csv, wobei * entweder borough, lsoa, msoa oder osward ist. Die Variablen area_sq_km und population enthalten die Information bzgl. Fläche und Bevölkerung.

- Laden Sie die Daten in R ein

- Überprüfen Sie, ob die Gesamtbevölkerung und Gesamtfläche gleich ist, egal welcher Aggregationsgrad verwendet wird. Bestimmen Sie die relativen prozentuellen Abweichungen vom jeweiligen Maximum.
- Erstellen Sie einen Dataframe in R, der Tabelle 1 in Aiello et al. (2020) reproduziert. Prüfen Sie, ob die exakt selben Ergebnisse reproduzierbar sind.
- Ergänzen Sie den Dataframe, indem Sie auch den Median der Bevölkerung und Fläche bestimmen.
- Visualisieren Sie die Verteilung von Bevölkerung und Fläche fur den kleinsten Aggregationsgrad.
- Interpretieren Sie die Ergebnisse.

Einlesen der Daten

```
yearLsoa <- read.csv(file = "year_lsoa_grocery.csv", header = TRUE, sep =",")
lsoaArea <- yearLsoa$area_sq_km
lsoaPopulation <- yearLsoa$population

yearMsoa <- read.csv(file = "year_msoa_grocery.csv", header = TRUE, sep =",")
msoaArea <- yearMsoa$area_sq_km
msoaPopulation <- yearMsoa$population

yearOsward <- read.csv(file = "year_osward_grocery.csv", header = TRUE, sep =",")
oswardArea <- yearOsward$area_sq_km
oswardPopulation <- yearOsward$population

yearBorough <- read.csv(file = "year_borough_grocery.csv", header = TRUE, sep = ",")
boroughArea <- yearBorough$area_sq_km
boroughPopulation <- yearBorough$population
```

Bestimmung der Gesamtbevölkerung, Gesamtfläche und relativen prozentuellen Abweichungen

```
areas <- list(lsoa = lsoaArea, msoa =  msoaArea,  ward = oswardArea, borough = boroughArea)
population <- list(lsoa = lsoaPopulation, msoa = msoaPopulation,
                   ward = oswardPopulation, borough = boroughPopulation)

lapply(areas, sum)
```

```
## $lsoa
## [1] 1571.78
##
## $msoa
## [1] 1572.48
##
## $ward
## [1] 1572.48
##
## $borough
## [1] 1572.48
```

```
lapply(population, sum)
```

```
## $lsoa
## [1] 8664108
##
## $msoa
## [1] 8666930
##
## $ward
## [1] 8666930
##
## $borough
## [1] 8666930
```

Nachdem die vier Aggregationsgrade eingelesen und nach der Fläche und Bevölkerung jeweils getrennt und gelistet wurden, haben wir mit lapply jeweils die Gesamtfläche und Gesamtbevölkerung bestimmt. Dabei wurde festgestellt, dass bis auf den Aggregationsgrad LSOA, alle gerade 1572.48km2 Fläche und 8666930 Einwohner haben. LSOA hat um 0,7 km2 weniger Fläche und um 2822 weniger Einwohner als die anderen. Die Abweichung ist minimal.

```
relDif <- function(var){
  x <- numeric()
  for (i in seq_along(var)){
    x <- append(x, round((max(var)-var[i])/max(var)*100,2))
  }
  return(x)
}

l <- lapply(areas, relDif)
lapply(l, head)
```

```
## $lsoa
## [1] 99.18 98.54 99.62 98.80 99.05 98.73
##
## $msoa
## [1] 87.08 90.37 90.46 88.95 94.70 92.25
##
## $ward
## [1] 95.66 95.32 95.56 88.36 88.12 95.01
##
## $borough
## [1] 98.07 75.96 42.21 59.65 71.17  0.00
```

```
l <- lapply(population, relDif)
lapply(l, head)
```

```
## $lsoa
## [1] 86.43 87.90 85.87 88.26 78.64 78.00
##
## $msoa
## [1] 63.38 59.60 41.30 64.21 49.39 47.08
##
## $ward
## [1] 47.96 60.73 49.82 60.71 58.91 54.60
##
## $borough
## [1] 98.24 46.56  0.34 36.23 14.90 14.41
```

Bei der Berechnung der jeweiligen relativen prozentuellen Abweichungen vom jeweiligen Maximum kann man aufgrund einiger Ausreißern bei allen Aggregationsgraden eine großteils hohe Abweichung vom Maximum feststellen.

Dataframe

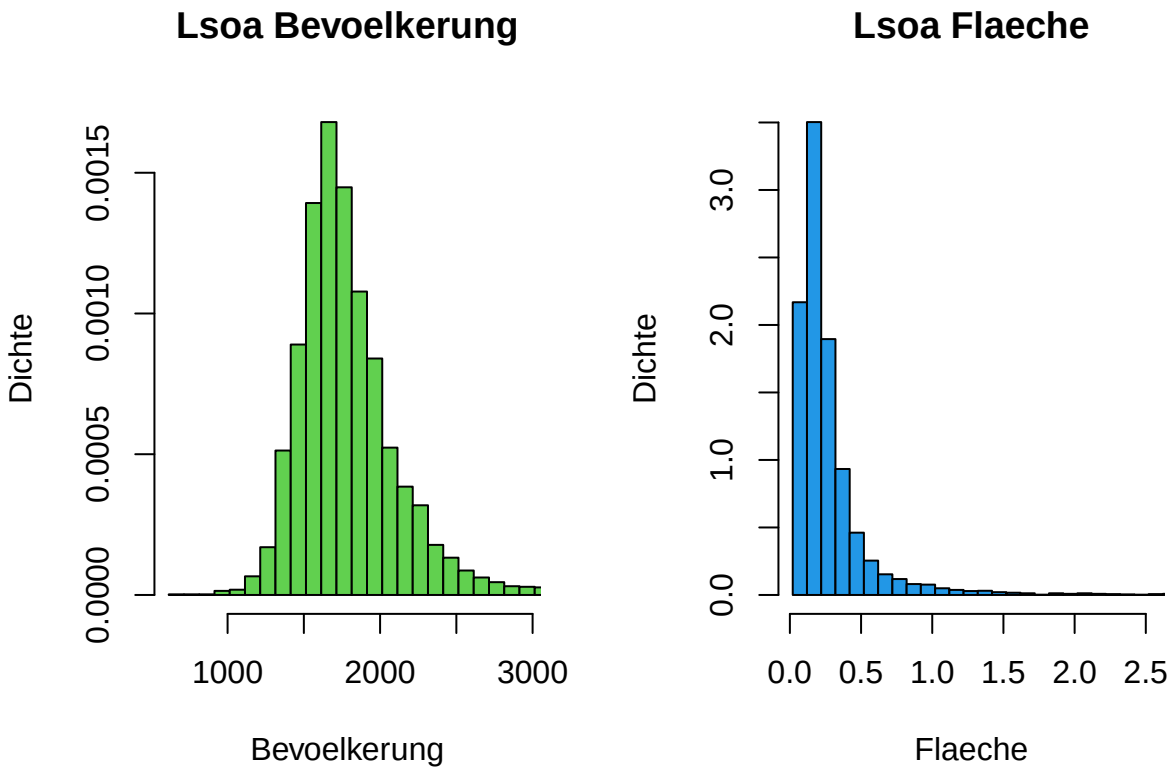
```
noAreas <- lapply(areas, length)

tab1Dataframe <- data.frame(
  Area = c("LSOA","MSOA","Ward","Borough"),
  NumberOfAreas = unlist(noAreas),
  AvgSurfaces = round(unlist(Map("/", lapply(areas, sum), noAreas)),2),
  AvgPopulation = round(unlist(Map("/", lapply(population, sum), noAreas)),0),
  MedianPopulation = unlist(lapply(population, median)),
  MedianArea = unlist(lapply(areas, median)),
  stringsAsFactors = FALSE
)
rownames(tab1Dataframe) <- NULL
tab1Dataframe
```

```
##      Area NumberOfAreas AvgSurfaces AvgPopulation MedianPopulation MedianArea
## 1    LSOA          4833        0.33          1793             1730       0.20
## 2    MSOA           983        1.60          8817             8587       1.15
## 3    Ward           638        2.46         13585            13637       1.82
## 4 Borough            33       47.65        262634           268626      37.64
```

Der Dataframe der Tabelle 1 in Aiello et al. (2020) ist annähernd der selbe wie der, den wir reproduziert haben. Einzig kleine Rundungsfehler können erkannt werden.

Visualisierung

```
par(mfrow=c(1,2))
hist(lsoaPopulation, main = "Lsoa Bevoelkerung", xlab = "Bevoelkerung",
     ylab = "Dichte",breaks= seq(min(lsoaPopulation),max(lsoaPopulation)+100, 100), col =3, freq = FALSE
hist(lsoaArea, main = "Lsoa Flaeche",xlab = "Flaeche", ylab = "Dichte",
     breaks= seq(min(lsoaArea),max(lsoaArea)+0.5, 0.1), col =4, freq = FALSE, xlim = c(min(lsoaArea),qu
```



Schlussendlich wurden die Daten des kleinsten Aggregationsgrades, also Lsoa, visualisiert. Um die Bevölkerung und Fläche zu visualisieren, haben wir ein Histogramm verwendet und uns dazu entschlossen die Daten bis zum 0,99-Quantil darzustellen, da die Veranschaulichung der Verteilung viel besser ist, wenn die Ausreißer nach oben gefiltert werden. Die Bevölkerung hat die meisten Ausprägungen bei ca 1800 und die Fläche zwischen 0.1 und 0.4 km2, was auch aus dem vorherigen Dataframe entnommen werden kann.

Aufgabe 2

Abbildung 3 in Aiello et al. (2020) visualisiert die Anzahl der Transaktionen auf LSOA Level.

- Entpacken Sie die Daten aus statistical-gis-boundaries-london.zip, sodass diese im Verzeichnis statistical-gis-boundaries-london sind.
- Laden Sie die Daten statistical-gis-boundaries-london/ESRI/LSOA_2011_London_gen_MHW.shp mithilfe von st_read aus dem Paket sf in R ein.
- Bestimmen Sie die Klasse des Objekts sowie die Dimensionen und die Spaltennamen.
- Verknüpfen Sie das Objekt mit den Daten aus year_lsoa_grocery.csv anhand der Variable LSOA11CD bzw. area_id. Bestimmen Sie die Klasse des Objekts sowie die Dimensionen und die Spaltennamen.
- Erstellen Sie die Grafik mit der Anzahl der Transaktionen. Achten Sie darauf, dass die Einteilung der Farbskala auf der logarithmierten Skala erfolgt und übergeben Sie eine passende Palette via pal.
- Erstellen Sie eine analoge Grafik mit der Anzahl der Transaktionen pro Einwohner.
- Vergleichen Sie und interpretieren Sie die Grafiken.

Einlesen der Daten

```
library(sf)
```

```
## Linking to GEOS 3.8.0, GDAL 3.0.4, PROJ 6.3.1
```

```
boundariesLondon <- st_read("./statistical-gis-boundaries-london/statistical-gis-boundaries-london/ESRI
```

Bestimmung von Klasse, Dimensionen und Spaltennamen

```
class(boundariesLondon)
```

```
## [1] "sf"         "data.frame"
```

```
dim(boundariesLondon)
```

```
## [1] 4835   15
```

```
colnames(boundariesLondon)
```

```
##  [1] "LSOA11CD"  "LSOA11NM"  "MSOA11CD"  "MSOA11NM"  "LAD11CD"   "LAD11NM"
##  [7] "RGN11CD"   "RGN11NM"   "USUALRES"  "HHOLDRES"  "COMESTRES" "POPDEN"
## [13] "HHOLDS"    "AVHHOLDSZ" "geometry"
```

Verknüpfung der Daten mit Lsoa

```
mergeboundaries <- merge(boundariesLondon, yearLsoa, by.x = "LSOA11CD", by.y = "area_id")
class(mergeboundaries)
```

```
## [1] "sf"         "data.frame"
```

```
dim(mergeboundaries)
```

```
## [1] 4833  216
```

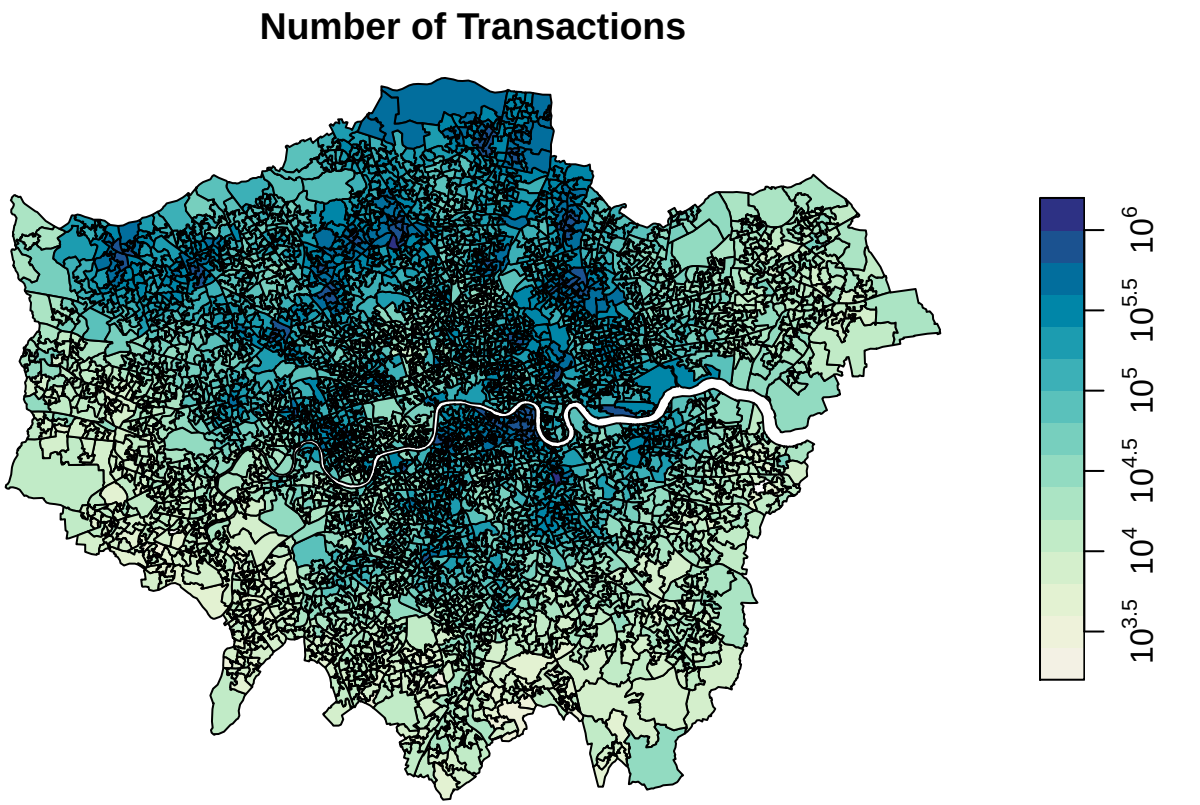
```
head(colnames(mergeboundaries))
```

```
## [1] "LSOA11CD" "LSOA11NM" "MSOA11CD" "MSOA11NM" "LAD11CD"  "LAD11NM"
```

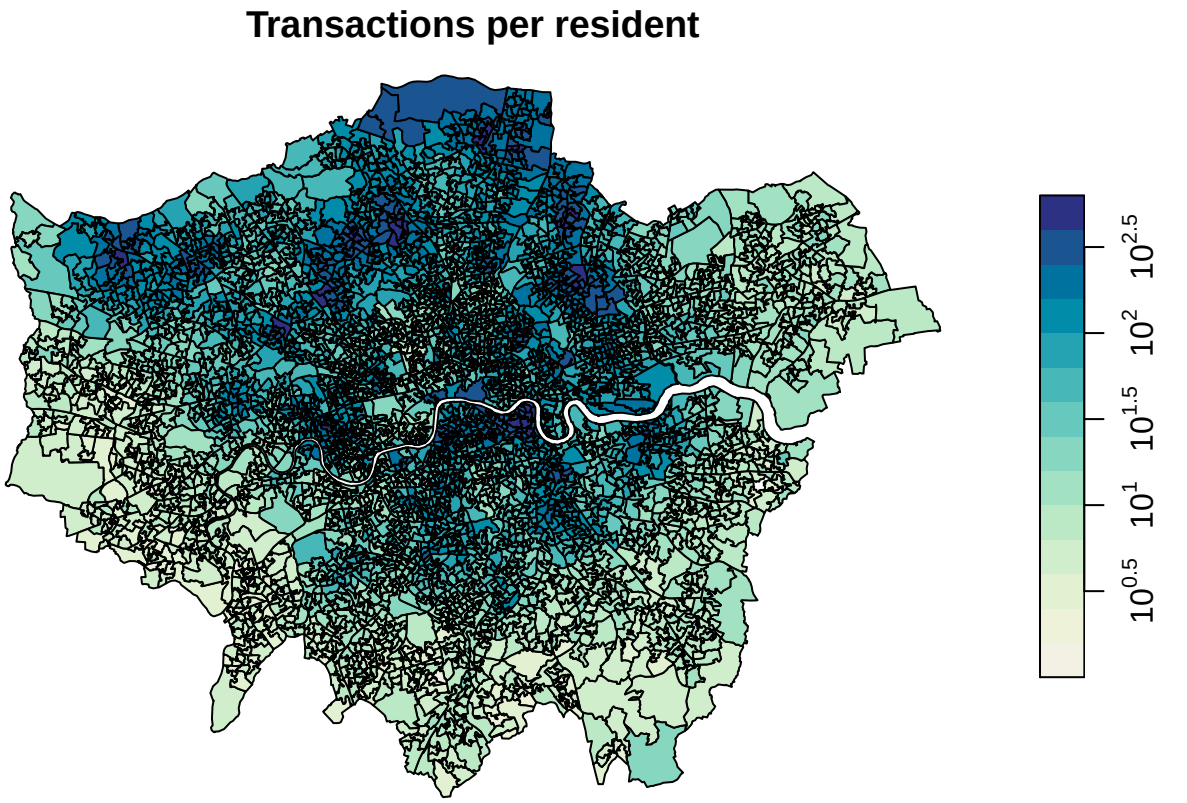
Man sieht, dass die Klasse beider Objekte gleich ist. Die eingelesene Datei besteht aus 4835 Zeilen und 15 Spalten. Dieses Objekt wird mit den LSOA-Daten verknüpft. Da diese allerdings weniger Beobachtungen beinhalten und bei der Verknüpfung ein sogenannter "inner-join" verwendet wird, hat auch das zusammengefügte Objekt weniger Beobachtungen (nämlich 4833). Die Variablen (Spalten) werden aneinandergereiht. Die Variable, nach der der Merge durchgeführt wurde, wird nicht dupliziert.

Visualisierung

```
library("colorspace")

par(mfrow=c(1,2))

mergeboundaries$perresident <- mergeboundaries$num_transactions/mergeboundaries$population

plot(mergeboundaries["num_transactions"], logz = TRUE, main = "Number of Transactions",
     pal= function(...) colorspace::sequential_hcl(...,palette = "Blue-Yellow", rev = TRUE))
```



```r
plot(mergeboundaries["perresident"], logz = TRUE, main = "Transactions per resident",
     pal= function(...) colorspace::sequential_hcl(...,palette = "Blue-Yellow",rev = TRUE))
```



Bei den beiden Grafiken lässt sich erkennen, dass die meisten Transaktionen (also Einkäufe) eher im Zentrum Londons getätigt wurden. Da die meisten Geschäfte ihren Standort wohl eher im Zentrum haben, ist das auch nachvollziehbar. Betrachtet man die Transaktionen pro Einwohner (Grafik 2), ist fast kein Unterschied zur ersten Grafik zu erkennen. Das liegt daran, dass die Anzahl der Einwohner auf LSOA-Level sehr gleichmäßig über alle LSOAs verteilt ist und sich dadurch die Anzahl der Transaktionen pro Einwohner in sehr ähnlichem verhältnis verändert.

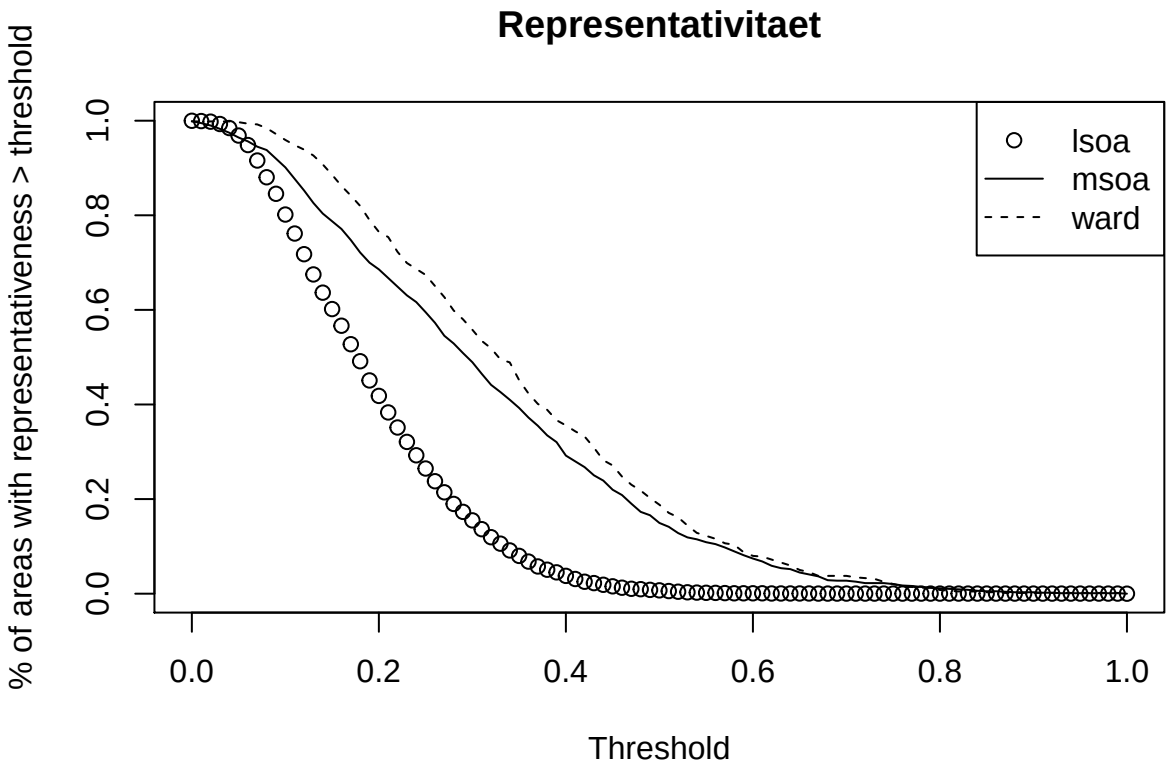
Aufgabe 3

Abbildung 4 links in Aiello et al. (2020) visualisiert den Prozentsatz an Fläche mit der normierten Repräsentativiät über einem Schwellwert gegen den Schwellwert.

- Verwenden Sie die Daten fur LSOA, MSOA und Ward, um die Grafik zu erzeugen.
- Vergleichen Sie Ihr Ergebnis mit der Grafik im Artikel und interpretieren Sie die Grafik.

Visualisierung

```r
sequ <- seq(0, 1, by = 0.01)

dat <- list(lsoa = yearLsoa, msoa = yearMsoa, ward = yearOsward)

repr <- function(var){
  x <- numeric()
  for(i in sequ){
    x <- append(x, nrow(var[var$representativeness_norm > i,])/nrow(var))
  }
  return(list(seq = sequ, perc = x))
}

normRepr <- lapply(dat, repr)

par(mfrow=c(1,1))
plot(as.data.frame(normRepr$lsoa), main = "Representativitaet", xlab = "Threshold",
     ylab = "% of areas with representativeness > threshold")
lines(as.data.frame(normRepr$msoa), lty = 1)
lines(as.data.frame(normRepr$ward), lty = 2)
legend("topright", c("lsoa", "msoa", "ward"), lty = c(NA,1,2), pch = c(1,NA,NA))
```



Bei dieser Grafik wird mithilfe der Daten von Lsoa, Msoa und Ward die Abbildung 4 links in Aiello et al. (2020) visualisiert. Wobei man bei dieser Grafik schon einen deutlichen Unterschied zu der Grafik in Aiello sieht. Bei Aiello sind die drei Aggregationsgrade annähernd ident und nähern sich bereits bei 0.4 Threshold der X-Achse. Bei unserer Grafik verhält sich nur Lsoa wie bei Aiello und Msoa und Ward flachen

erst ab 0.5 ab und schneiden erst bei 0.8 die X-Achse.

Aufgabe 4

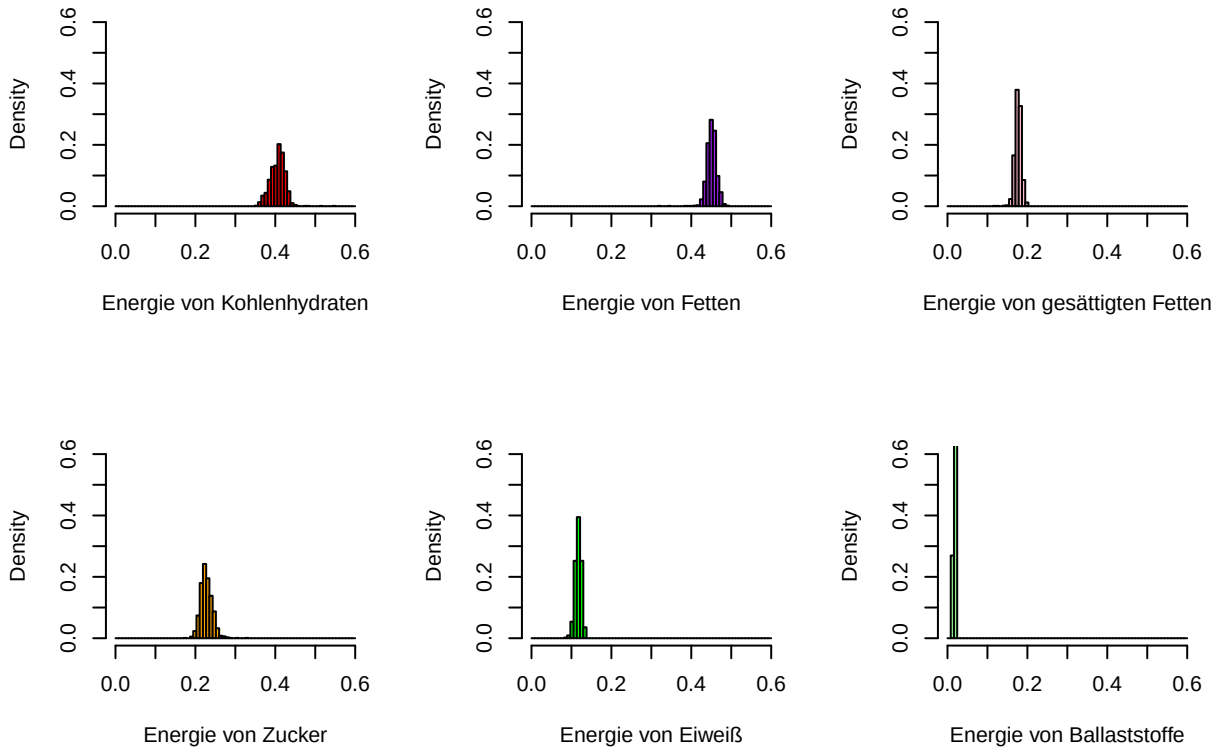
Abbildung 1 in Aiello et al. (2019) stellt die Verteilung der relativen Energie pro Nährstoff an der Gesamtenergie dar.

- Verwenden Sie die Daten fur MSOA um die Histogramme für die 6 Nährstoffe zu erstellen. Achten Sie auf passende Achsenbeschriftungen und stellen Sie sicher, dass die x- und y-Achsengrenzen sowie die Intervalleinteilung bei allen Histogrammen gleich sind.
- Interpretieren Sie die Grafik.

Visualisierung

```
energyTot <- yearMsoa$energy_tot
energyFatTot <- yearMsoa$energy_fat / energyTot
energyFibreTot <- yearMsoa$energy_fibre / energyTot
energySatTot <- yearMsoa$energy_saturate / energyTot
energyCarbTot <- yearMsoa$energy_carb / energyTot
energySugarTot <- yearMsoa$energy_sugar / energyTot
energyProteinTot <- yearMsoa$energy_protein / energyTot

par(mfrow=c(2,3))

h <- hist(energyCarbTot, breaks=seq(0,0.6,length=75), plot = F)
h$density <- h$counts/sum(h$counts)
plot(h, main = NULL, xlab = "Energie von Kohlenhydraten",xlim = c(0,0.6),
     ylim = c(0,0.6), col = "red", freq = F)

h <- hist(energyFatTot, breaks=seq(0,0.6,length=75), plot = F)
h$density <- h$counts/sum(h$counts)
plot(h, main = NULL, xlab = "Energie von Fetten",xlim = c(0,0.6),
     ylim = c(0,0.6), col = "purple", freq = F)

h <- hist(energySatTot, breaks=seq(0,0.6,length=75), plot = F)
h$density <- h$counts/sum(h$counts)
plot(h, main = NULL, xlab = "Energie von gesättigten Fetten",xlim = c(0,0.6),
     ylim = c(0,0.6), col = "pink", freq = F)

h <- hist(energySugarTot, breaks=seq(0,0.6,length=75), plot = F)
h$density <- h$counts/sum(h$counts)
plot(h, main = NULL, xlab = "Energie von Zucker",xlim = c(0,0.6),
     ylim = c(0,0.6), col = "orange", freq = F)

h <- hist(energyProteinTot, breaks=seq(0,0.6,length=75), plot = F)
h$density <- h$counts/sum(h$counts)
plot(h, main = NULL, xlab = "Energie von Eiweiß",xlim = c(0,0.6),
     ylim = c(0,0.6), col = "green", freq = F)
```

```
h <- hist(energyFibreTot, breaks=seq(0,0.6,length=75), plot = F)
h$density <- h$counts/sum(h$counts)
plot(h, main = NULL, xlab = "Energie von Ballaststoffe",xlim = c(0,0.6),
     ylim = c(0,0.6), col = "lightgreen", freq = F)
```



Bei diesen 6 Histogrammen haben wir die Energie der 6 Nährstoffe dargestellt. Auf der Y-Achse ist die Dichte und auf der X-Achse die Energie abgebildet. Die Londoner kaufen am meisten Produkte mit Fett und Kohlenhydraten. Am wenigsten werden Ballaststoffe eingekauft.

Aufgabe 5

Die Datei diabetes_estimates_osward_2016.csv enthält Gesundheitsinformationen auf Ward Level. Die Ward Level Information ist auch enthalten in statistical-gis-boundaries-london/ESRI/London_Ward_CityMerged.shp bzw. London-wards-2014/London-wards-2014_ESRI/London_Ward_CityMerged.shp. Schreiben Sie eine Funktion, die die Erhebungseinheiten in x mit den Erhebungseinheiten y anhand von id.x bzw. id.y verknüpft und folgende Information zurückgibt: wie viele Einheiten sind sowohl in x als auch in y, sind nur in x bzw. nur in y, wendet FUN auf die Variablen var.x und var.y an je nach Gruppierung, ob die Erhebungseinheit sowohl in x als auch in y ist, nur in x bzw. nur in y ist.
> compare_data <- function(x, y, id.x, id.y = id.x, var.x = character(0), + var.y = var.x, FUN = sum, ...) { + ... + }

- Überprüfen Sie die Argumente und geben Sie passende Fehlermeldungen aus. Das ... Argument soll der Funktion FUN ubergeben werden.
- Geben Sie einen Dataframe zurück, der die drei Gruppen in den Zeilen und die Anzahl sowie das Ergebnis von FUN jeweils in den Spalten hat.
- Wenden Sie die Funktion an, indem Sie x gleich den Daten in statistical-gis-boundaries-london/ESRI/London_Ward_CityMerged.shp setzen und y gleich den Daten in London-wards-2014/London-wards-2014_ESRI/London_Ward_CityMerged.shp setzen. Bestimmen Sie die passenden ID Variablen und wählen Sie die Fläche jeweils als Variable.
- Wenden Sie die Funktion an, indem Sie x gleich den Daten in diabetes_estimates_osward_2016.csv setzen und y gleich den Daten in London-wards-2014/London-wards-2014_ESRI/London_Ward_CityMerged.shp setzen. Bestimmen Sie die passenden ID Variablen und wählen Sie die Fläche als Variable.
- Wenden Sie weiters die Funktion an, indem Sie x gleich den Daten in diabetes_estimates_osward_2016.csv setzen und y gleich den Daten in year_osward_grocery.csv setzen. Bestimmen Sie die passenden ID Variablen und wählen Sie die Fläche und Bevölkerung als Variablen.
- Interpretieren Sie die Ergebnisse.

Einlesen der Daten

```
diabetes <- read.table("diabetes_estimates_osward_2016.csv", header = TRUE, sep = ",")
head(diabetes)

londonWardX <- st_read("./statistical-gis-boundaries-london/statistical-gis-boundaries-london/ESRI/Londo
londonWardY <- st_read("./London-wards-2014/London-wards-2014 (1)/London-wards-2014_ESRI/London_Ward_Ci
```

Funktion

```
compare_data <- function(x, y, id.x, id.y = id.x, var.x = character(0),
                         var.y = var.x, FUN = sum, ...) {
  # Zwei Listen müssen übergeben werden, ansonsten nicht genügend Daten
  if(!is.list(x) | !is.list(y))
    stop("Bitte überprüfen Sie Ihre Eingabe für x und y!")
  # ID muss als String übergeben werden (zumindest x, y nicht unbedingt.
  # Wenn y übergeben wird, muss es auch ein String sein)
  if(!is.character(id.x) | ((id.x != id.y) & !is.character(id.y)))
    stop("es muss eine korrekte ID angegeben werden!")
  # Variable muss als String übergeben werden (zumindest x, y nicht unbedingt.
  # Wenn y übergeben wird, muss es auch ein String sein)
  if(!is.character(var.x) | ((var.x != var.y) & !is.character(var.y)))
    stop("es muss eine korrekte Variable angegeben werden!")


  #spezielle Behandlung bei merge von 2 sf Objekten
  if(class(x)[1] == "sf" & class(y)[1] == "sf"){
    # verknüpft werden die Objekte, indem sie zuerst in einen Dataframe umgewandelt werden
    # und mit der jeweiligen ID verbunden werden. Dabei wird
    # sozusagen die Geometry-Variable deaktiviert
    mergedXandY <- merge(x %>% as.data.frame(), y %>% as.data.frame(),
                         by.x = id.x, by.y = id.y)
    # Danach wird die Geometry-Variable wieder "aktiviert" und wieder in ein Objekt
    # der Klasse sf umgewandelt
```

```
    mergedXandY %>% st_sf(sf_column_name = 'geometry.x')
    mergedX <- merge(x %>% as.data.frame(), y %>% as.data.frame(),
                     by.x = id.x, by.y = id.y, all.x = TRUE)
    mergedX %>% st_sf(sf_column_name = 'geometry.x')
    mergedY <- merge(x %>% as.data.frame(), y %>% as.data.frame(),
                     by.x = id.x, by.y = id.y, all.y = TRUE)
    mergedY %>% st_sf(sf_column_name = 'geometry.y')

    # Funktion wird auf die einzelnen Gruppierungen angewendet
    sumXandY <- sumXandY1 <- (FUN(get(paste(var.x, ".x",sep = ""), mergedXandY), ...))
    sumXandY2 <- (FUN(get(paste(var.y, ".x",sep = ""), mergedXandY), ...))
    sumX <- (FUN(get(paste(var.x,".x",sep = ""), mergedX), ...))
    sumY <- (FUN(get(paste(var.y,".y",sep = ""), mergedY), ...))
  }
  else{
    # Verknüpfung der Objekte nach den jeweiligen IDs
    mergedXandY <- merge(x, y, by.x = id.x, by.y = id.y)
    mergedX <- merge(x, y, by.x = id.x, by.y = id.y, all.x = TRUE)
    mergedY <- merge(x, y, by.x = id.x, by.y = id.y, all.y = TRUE)

    sumXandY <- sumXandY1 <- (FUN(get(var.x, mergedXandY), ...))
    sumXandY2 <- (FUN(get(var.y, mergedXandY), ...))
    sumX <- (FUN(get(var.x, mergedX), ...))
    sumY <- (FUN(get(var.y, mergedY),...))
  }

  # wenn die Ergebnisse der angewendeten Funktion aus dem ersten für die erste und zweite
  # Variable nicht übereinstimmen, wird für das Ergebnis der gemeinsamen Daten
  # von X und Y NA als Ergebnis verwendet
  if(sumXandY1 != sumXandY2)
    sumXandY <- NA

  df <- data.frame(mergeType = c("X und Y", "nur X", "nur Y"),
                   anzahl = c(nrow(mergedXandY),nrow(mergedX)-nrow(mergedXandY),
                              nrow(mergedY)-nrow(mergedXandY)),
                   ergebnis = c(round(sumXandY,0), if(var.x %in% colnames(x)) round(sumX-sumXandY1,0)
                                else NA,
                                if(var.y %in% colnames(y)) round(sumY-sumXandY2,0) else NA))

  return(df)
}
```

Anwendung der Funktion

```
compare_data(londonWardX, londonWardY, "GSS_CODE", "GSS_CODE", "HECTARES", "HECTARES", na.rm = TRUE)
```

```
##   mergeType anzahl ergebnis
## 1   X und Y    571   154169
## 2     nur X     54     5301
## 3     nur Y     59     5301
```

Die beiden Datensätze verfügen jeweils einige Beobachtungen, die im anderen Datensatz nicht vorkommen. Die Fläche dieser einzigartigen Beobachtungen summiert sich allerdings bei beiden auf exakt das gleiche. Nimmt man die Tabelle aus Aufgabe 1 heran und rechnet die Gesamtfläche für das Level Ward hoch, kommt man auch auf einen ähnlichen Wert.

```
compare_data(diabetes, londonWardY, "area_id", "GSS_CODE", "HECTARES", na.rm= TRUE)
```

```
##   mergeType anzahl ergebnis
## 1   X und Y    629   159155
## 2     nur X      8       NA
## 3     nur Y      1      315
```

Hier sollten die Gesundheitsinformationen auf Ward Level mit den generellen Daten über dieses Level verknüpft werden. Es lässt sich erkennen, dass nur wenige IDs der Datensätze nicht übereinstimmen. Man sollte die Fläche als Variable heranziehen, allerdings kommt keine Variable für die Fläche im ersten Datensatz vor, weshalb dadurch auch keine Auswertung zustand kam. 315 Hektar beträgt die Fläche der Beobachtung, die nur im zweiten Datensatz vorkommt.

```
compare_data(diabetes, yearOsward, "area_id", "area_id", "area_sq_km", na.rm = TRUE)
```

```
##   mergeType anzahl ergebnis
## 1   X und Y    547     1340
## 2     nur X     90       NA
## 3     nur Y     91      233
```

Nun wird die Gesundheitsinformationen auf Ward Level mit den Einkaufs- bzw. Lebensmitteldaten auf Ward Level verbunden. Zuerst soll die Fläche betrachtet werden. Hier ist die Einheit der Fläche in km2 und nicht in Hektar! Es gibt wieder einige verschiedene IDs. Auch hier kann für die Erhebunseinheiten, die nur in X vorkommen keine Fläche berechnet werden, da keine entsprechende Variable in diesem Datensatz vorkommt. Die Fläche der Erhebungseinheiten, die nur in Y vorkommen, beläuft sich auf 233 km2. Addiert man diesen Wert mit der Fläche der Beobachtungen, die in X und in Y enthalten sind und wandelt man diesen Wert in Hektar um (mal 100), erhält man wieder ein ähnliches Ergebnis, das mit der Tabelle aus Aufgabe 1 übereinstimmt.

```
compare_data(diabetes, yearOsward, "area_id", "area_id", "gp_patients", "population", na.rm = TRUE)
```

```
##   mergeType anzahl ergebnis
## 1   X und Y    547       NA
## 2     nur X     90  1054849
## 3     nur Y     91  1229975
```

Zu den selben Daten soll auch noch die Bevölkerung betrachtet werden. Natürlich ist die Anzahl der Elemente in den jeweiligen Gruppierungen wieder gleich wie in der vorherigen Auswertung, da die gleichen Datensätze verwendet wurden. Nun kann man auch ein Ergebnis für die Erhebungseinheiten in X berechnen, genauso wie für Y. Allerdings wird hier kein Ergebnis für die Erhebunseinheiten, die sowohl in X als auch in Y vorkommen, ermittelt. Die Werte der beiden Variablen stimmen für die einzelnen Erhebungseinheiten nicht überein. Während sich die eine Variable auf die Anzahl der Patienten bezieht, beschreibt die andere Variable die ganze Bevölkerung auf Ward Level. Dadurch würden unterschiedliche Resultate entstehen, je nachdem welche Variable herangezogen werden würde, wodurch wir uns entschieden haben, keine Resultat zu ermitteln.

Aufgabe 6

Verknüpfen Sie die Daten aus year_osward_grocery.csv mit der Information in diabetes_estimates_osward_2016.csv.

- Bestimmen Sie mithilfe Spearman Korrelation, ob es einen Zusammenhang zwischen der geschätzten Diabetes Prävalenz und der Energie der Nährstoffe gibt.
- Visualisieren Sie in einem Streudiagramm den Zusammenhang zwischen der geschätzten Diabetes Prävalenz und der Energie der Nährstoffe.
- Interpretieren Sie das Ergebnis. Gibt es empirische Evidenz, dass sich das Kaufverhalten im Supermarkt auf die geschätzte Diabetes Prävalenz auswirkt.

Verknüpfung der Daten

```
merged <- merge(yearOsward, diabetes, by= "area_id")
```

Berechnung des Zusammenhangs

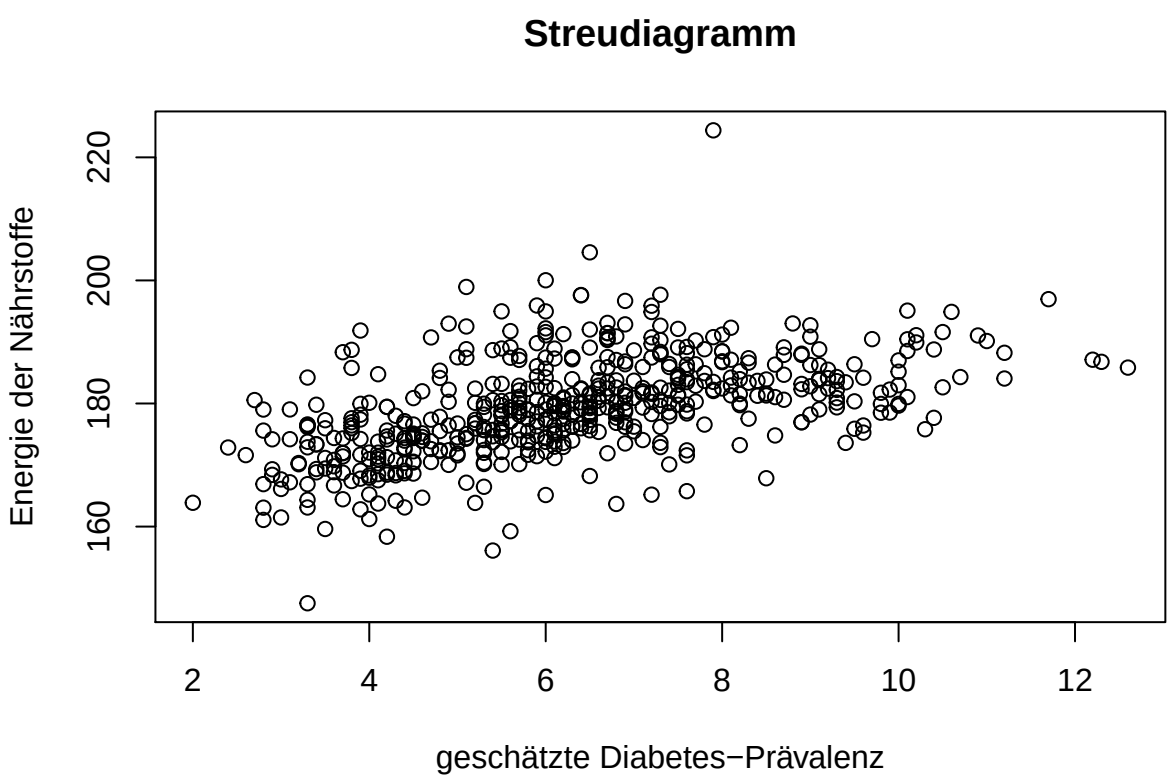
```
cor.test(merged$estimated_diabetes_prevalence, merged$energy_tot, method = "spearman")
```

```
## Warning in cor.test.default(merged$estimated_diabetes_prevalence,
## merged$energy_tot, : Cannot compute exact p-value with ties
```

```
##
##  Spearman's rank correlation rho
##
## data:  merged$estimated_diabetes_prevalence and merged$energy_tot
## S = 11343730, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.5841405
```

Visualisierung

```
par(mfrow=c(1,1))
plot(merged$estimated_diabetes_prevalence, merged$energy_tot, main = "Streudiagramm",
     xlab = "geschätzte Diabetes-Prävalenz", ylab = "Energie der Nährstoffe")
```



Bei dieser Aufgabe wurden die Daten aus year_osward_grocery.csv mit der Information in diabetes_estimates_osward_2016.csv. verknüpft und die Korrelation zwischen der geschätzten Diabetes Praevalenz und der Energie der Nährstoffe berechnet und mit einem Streudiagramm visualisiert. Die Korrelation mithilfe von Spearman ist 0.58, also gibt es einen mittleren gleichsinnigen Zusammenhang der beiden Variablen. Das heißt, in Regionen mit einer höheren Diabetes-Prävalenz enthält das durchschnittliche Produkt mehr Energie - das Kaufverhalten hat also einen Einfluss. Dieses Ergebnis bestätigt das Streudiagramm, in dem sich dieser Zusammenhang auch erkennen lässt.