Investigating the risk factors associated with BMI: EasyShare data

Max Guy, Katrina Sanders, Kim Johnston

Main Report

This investigation was carried out equally by the following people with their responsibilities: Katrina Sanders, s1901708, report write up and model building, Kim Johnston, s1936036, report write up, Max Guy, s1945362, writing the code functions.

Our report explores risk factors of obesity using the easySHARE data. The data was collected over many years, based mostly in Europe, about all aspects of life where individuals, couples, and households filled out questionnaires. The main topics of this data were surrounding health, such as social support, childhood conditions, and functional limitation indices. The data has 107 variables with 412110 observations, observed over 8 waves of surveys over 10 years. However, we found the data contained many missing values due to a range of reasons (see Appendix C). This report analyses the data to be able to make suitable conclusions on the following questions: 1. Which of the variables are associated with BMI (an indicator of obesity)? 2. Are the associations between risk factors and obesity the same for males and females?

Our report also considers the strength and nature of the associations between risk factors.

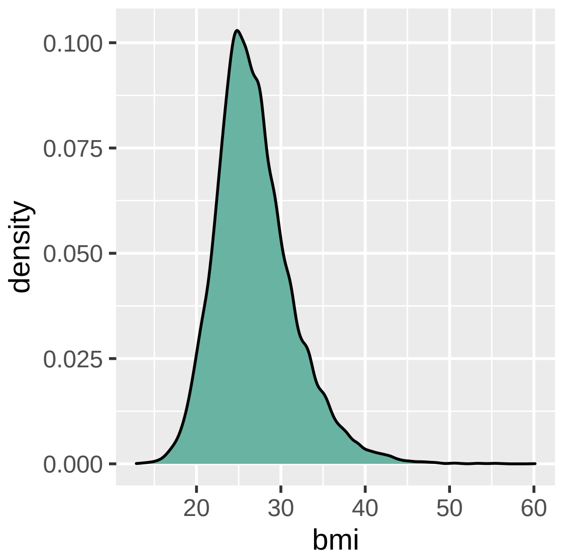
In order to begin our analysis, we chose to work with a subset of the data covering 1 wave and 4 countries. We chose wave 6 along with countries 15, 16, 23, and 25 (which represent Spain, Italy, Belgium, and Israel respectively) to use for our model. The decision process for this is detailed in Appendix A. Wave 6 was carried out in 2015 so the data is still fairly current as well as not being skewed by COVID-19.

During the data cleaning process, we found columns with a large number of "NA" values so decided that any column containing more than 15% "NA" should be removed. This resulted in removing 23 columns. Further data cleaning had us remove "year", "wave number", "mergeid", "hhid", and "coupleid".

We then started a very high level analysis on our subset od data and looked at summary statistics. The first thing we noted of importance was the vast number of variables and observations, in total there are 84 variables, and 18,769 observations.

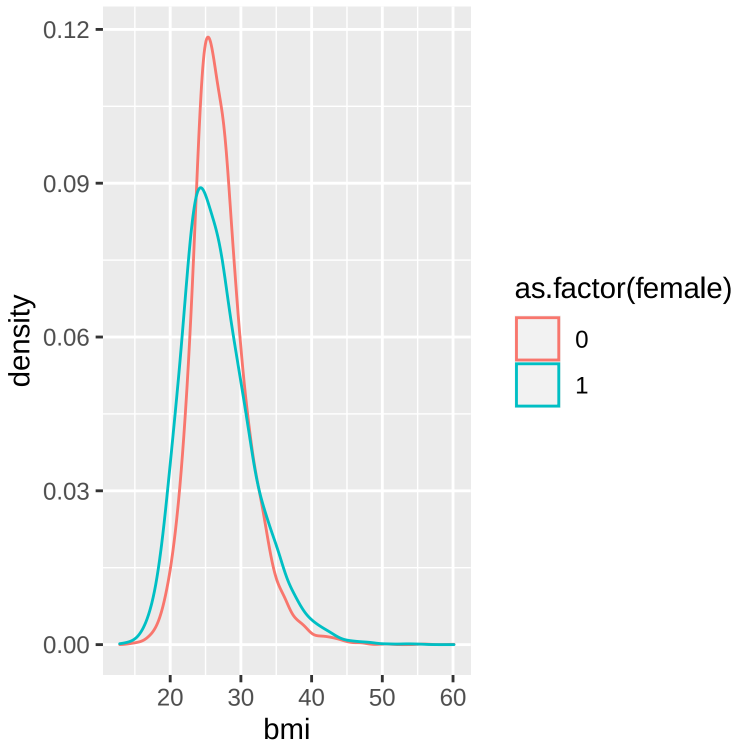
As we wanted to focus on the factors associating to bmi, we looked at a density plot and found that the data roughly followed a normal distribution, this showed our dependent variable wasn't skewed and was unimodal (mode between 20-30).

```
## Warning: Removed 800 rows containing non-finite values (stat_density).
```



We also explored the difference in BMI between males and females. This plot showed that males have a BMI much more concentrated around their mean, however females have a wider spread of BMI values. Something which may be of significance in our linear model.

```
## Warning: Removed 800 rows containing non-finite values (stat_density).
```



We decided to take a generative approach to building our model (see Appendix B) where we correlated all the variables not yet in our model individually against bmi and added the most suitable variable to the model. We then evaluated this new model and evaluated the place of the new variable in the model using F-tests and by looking at p-values. If we concluded that the new variable was a good addition to the model, we would repeat this iterative process. The process terminates when adding new variables to the model no longer improves the model. This was determined by the R^2 value and when adding new variables was no longer significant.

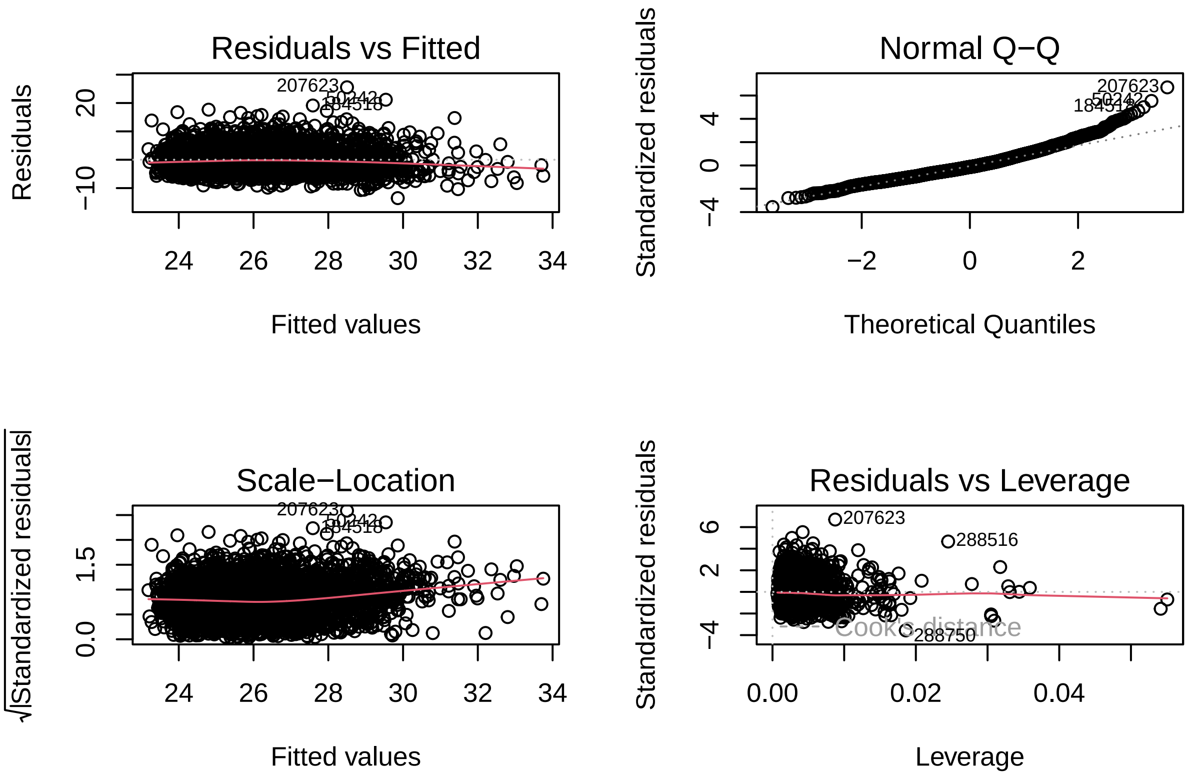
Some interesting anomalies we encountered while building the model: When adding the "sphus" variable to the model, it failed our statistical tests (p-value = 0.66), so it was discarded from the model. We noticed high correlations between this variable and others already in the model which may explain the high p-value. Adding "sp00_3_3_mod" to the model resulted in a spike in r^2 to 0.2043 however further analysis revealed that due to too many "NA" values in the model, our number of datapoints for which the model is based upon decreased to less than 100 observations (from an initial almost 19,000 observations) and so it was also rejected as a new predictor. At iteration 13, we had to remove the "gender_partner" variable as the new "female" variable was performing better in the model and there is an interaction between the 2 variables so we can;t include both in our model. We finally decided to stop at iteration 14, as adding the next most correlated variable "sp003_2_mod" resulted in a decreased r^2 as well as increased p-values.

Therefore, our final r^2 value is 0.1401, represented by our final linear model:

$$\begin{aligned}\widehat{\text{bmi}} = 26.1 + 0.91(\text{chronic\_mod}) + 0.53(\text{lgmuscle}) + 1.6(\text{mobilityind}) - \\ 0.08(\text{eduyears\_mod}) - 0.3(\text{co007\_}) - 1.41(\text{grossmotor}) + 0.12(\text{ch001\_}) + \\ 0.03(\text{casp}) + 0.09(\text{euro9}) - 1.41(\text{female}) + 0.01(\text{ep013\_mod})\end{aligned} \tag{1}$$

From our linear model, we found the 3 most influential variables on bmi to be "female", "mobilityind", "grossmotor". Both "grossmotor" and "female" have estimates of around -1.4 meaning they reduce "bmi", in essence this means that being female results in a lower bmi than being male - which can be seen in our initial analysis. However, unexpectedly this suggests that finding motor activities easier results in a higher "bmi". Then "mobilityind" had the biggest positive effect on "bmi", meaning the less mobile someone is, the higher their "bmi". These are unexpected results as "mobilityind" and "grossmotor" are very similar variables, so we would expect them to have similar effects on bmi.

Finally, we tested the assumptions of our model by creating these plots, and making the following conclusions:



Residuals vs Fitted: The residuals "bounce randomly" around the residuals = 0 line, suggesting that the assumption that the relationship is linear is reasonable. They also roughly form a horizontal band around this line, suggesting that the variances of the error terms are equal. Finally, no residuals stand out from the basic random pattern, suggesting that there are no outliers.

Normal Q-Q: From this plot we can't be sure to assume normality since it shows a right skew. Therefore, this assumption may contribute to our low confidence in the model.

Scale-Location: The red line is approximately horizontal so the average magnitude of the standardised residuals isn't changing much across all fitted values and the spread around the red line doesn't vary much, so the variability is fairly constant.

Residuals vs Leverage: This plot helps us to find influential cases, looking out for outlying values. Since the majority of our points have low leverage they have a weak influence on the coefficients in the regression model.

The other question we wanted to answer regarding the difference in male and females was: How does the model differ if fitted on only female data vs only male data?

Looking at the difference between models fitted on only male data vs only female data. Note, this model is identical to the general model with the obvious exception; the gender predictor is no longer included. The main interesting differences are: Males have a higher base intercept which agrees with the previous coefficient conclusion. There is a notable difference in the adjusted R^2 values of the models with the females having an R^2 of 0.1565 and the males just 0.0671, implying the predictions made for females may be generally more accurate.

Overall, although we have constructed a linear model, we have little confidence in the conclusions it produces. The unexpected conclusions around "grossmotor" and "mobilityind" adds to our reduced confidence. The main reason for this is the low r^2 value, which suggests our variables in the model only explains 14% of the variability in bmi.

Executive Summary

The investigation we carried out was based on the easySHARE dataset which was collected over many years, based mostly in Europe, where individuals, couples, and households filled out questionnaires about all aspects of life. From this data, we investigated the factors which are associated with bmi and if certain risk factors are the same for both male and female. We took a subset of this data to focus on and from this drew conclusions about it.

The 2 main factors we found which have the largest effect on bmi were variables indicating gender and the mobility of individuals. Our report concludes that just by being female bmi is lowered by 1.4, backing up pre assumed beliefs that females have a lower bmi than males. This is highlighted in this box plot, where the mean for females is seen to be lower. We also found that the more mobility issues an individual had, especially regarding physical activities, resulted in a higher bmi. The mobility issues were scored in 4 levels, in increments of 1, with 0 being most mobile and 4 being least mobile. With each level we found bmi to increase by 1.6, so for those scoring 4 are likely to have a bmi 6.4 times higher than those scoring 0. The lowered mobility may be resulting in less exercise for these individuals and therefore increasing their bmi.

We also took a closer look at the difference between factors affecting bmi for males vs females. We again found that females have a lower bmi on average, as well as overall the factors we found to affect bmi are a more accurate representation for females. This suggests that there may be other factors we haven't looked at that affect bmi for males more.

In conclusion, although gender and mobility seemed to have an effect on bmi we couldn't make solid conclusions from our data or analysis. We found the variables we had only explained around 14% of factors affecting bmi, meaning there are other variables which affect bmi which we didn't include in the analysis we carried out on chosen variables. Therefore, to make stronger conclusions we would need more data, or more variables which would explain the reasons for variability in bmi of an individual.

Appendix A - quality_check.R

```r
#load data and libraries
library(tidyverse)

load("data/easySHARE_rel8_0_0.rda")
df <- easySHARE_rel8_0_0

#select numeric columns
bool_df <- df %>% select_if(is.numeric)

#maps negative numbers to 0 and all other numbers to 1
functional_approach <- function(x){
  return(floor(sqrt(sign(x)+1)))
}

#make df of booleans
bool_df <- map_df(bool_df, .f=functional_approach)

#replace necessary information columns
bool_df$wave <- df$wave
bool_df$country <- df$country

selected_waves <- c(6,7)
selected_countries <- c(15,16,23,25)

#get number of missing values in each wave-country combo
score_combo <- function(dataframe, wavenum, countrynum){
  df_total <- dataframe %>%
    filter(wave == wavenum) %>%
    filter(country == countrynum) %>%
    select(-c(wave, country)) %>%
    mutate(sum = rowSums(.))

  total <- sum(df_total$sum)
  possible_total <- dim(df_total)[1] * dim(df_total)[2]

  quality <<- total/possible_total
  return(quality)
}

#get proportion of data present in each wave-country combination
for(i in selected_waves){
  for(j in selected_countries){
    quality <- score_combo(bool_df, i, j)
    print(i)
    print(j)
    print(quality)
    print(" ")
  }
}

#make smaller dataframe on subset we want to analyse
small_bool <- bool_df %>%
  filter(wave %in% selected_waves) %>%
  filter(country %in% selected_countries)


#make function to give info on how useful a wave is
score_wave <- function(dataframe, wave_num){
  temp_df <- dataframe %>%
    filter(wave == wave_num)

  length = dim(temp_df)[1]
  totals <- c()
  colnames <- colnames(temp_df)
  cols <- c()
  for(i in 1:ncol(temp_df)){
    total <- sum(temp_df[i])
    print(total)
    totals <- append(totals, total)
  }
  props <- totals/length

  viable_columns <- c()
  for(i in 1:length(props)){
    viable <- 0
    if(props[i] > 0.85){
      viable <- 1
    }
    viable_columns <- append(viable_columns, viable)
  }
  df <- data.frame(colnames, props, viable_columns)
  return(df)
}


#make function to give info on how useful a country is
score_country <- function(dataframe, country_num){
  temp_df <- dataframe %>%
    filter(country == country_num)

  length = dim(temp_df)[1]
  totals <- c()
  colnames <- colnames(temp_df)
  cols <- c()
  for(i in 1:ncol(temp_df)){
    total <- sum(temp_df[i])
    print(total)
    totals <- append(totals, total)
  }
  props <- totals/length

  viable_columns <- c()
  for(i in 1:length(props)){
    viable <- 0
    if(props[i] > 0.85){
      viable <- 1
    }
    viable_columns <- append(viable_columns, viable)
  }
  df <- data.frame(colnames, props, viable_columns)
  return(df)
}



wave_6 <- score_wave(small_bool, 6)
wave_7 <- score_wave(small_bool, 7)

country_15 <- score_country(small_bool, 15)
country_16 <- score_country(small_bool, 16)
country_23 <- score_country(small_bool, 23)
country_25 <- score_country(small_bool, 25)

w6_num_viable <- sum(wave_6$viable_columns)
w7_num_viable <- sum(wave_7$viable_columns)

c15_num_viable <- sum(country_15$viable_columns)
c16_num_viable <- sum(country_16$viable_columns)
c23_num_viable <- sum(country_23$viable_columns)
c25_num_viable <- sum(country_25$viable_columns)


print(paste(w6_num_viable, w7_num_viable))
print(paste(c15_num_viable, c16_num_viable, c23_num_viable, c25_num_viable))

#conclusion: use wave 6 only but use all 4 countries


#wave 6 investigation
small_bool <- small_bool %>%
  filter(wave ==6)

country_15 <- score_country(small_bool, 15)
country_16 <- score_country(small_bool, 16)
country_23 <- score_country(small_bool, 23)
country_25 <- score_country(small_bool, 25)

c15_num_viable <- sum(country_15$viable_columns)
c16_num_viable <- sum(country_16$viable_columns)
c23_num_viable <- sum(country_23$viable_columns)
c25_num_viable <- sum(country_25$viable_columns)

print(paste(c15_num_viable, c16_num_viable, c23_num_viable, c25_num_viable))
```

Appendix B - model_functions.R

```
#Functions for making a linear model generatively
```

```r
#import libraries
library(tidyverse)

#quick function to assess quality of dataframe
negativeNumberMapping <- function(x){
  return(floor(sqrt(sign(x)+1)))
}

#remove columns where >15% of values are nan
removeBadColumns <-function(df){
  #turn all negative numbers into nan values
  bool_df <- map_df(df, .f=negativeNumberMapping)

  #get length of dataframe for finding % good data entries
  length_df <- dim(df)[2]
  lockBinding("length_df", environment())

  #create empty vector of columns to remove
  cols_to_remove <- c()

  #get proportion good data values. If proportion too low, add to list to be removed
  for(i in 1:dim(df)[2]){
    num_useful <- sum(bool_df[i])
    prop_numeric <- num_useful/length_df
    if(prop_numeric < 0.85){
      cols_to_remove <- append(cols_to_remove, i)
    }
  }

  #remove low quality columns from dataframe
  df <- df[-cols_to_remove]

  #remove wave and year columns
  df <- df %>%
    select(-c(wave, int_year))

  return(df)
}


#replace all negative numbers with nan values
#this function takes a while unfortunately :(
setNanValues <- function(df){
  for(i in 1:dim(df)[1]){
    for(j in 1:dim(df)[2]){
      if(df[i,j] < 0){
        df[i,j] = NA
      }
    }
  }
  return(df)
}
```

```r
#initialise analysis
initialiseSearch <- function(df){

  #make global vector for storing evaluations
  evaluations <<- c(0)

  #make global log for model construction process
  sink("output.txt")
  cat("Log Initialised")
  cat("\n")
  sink()

  #initialise iteration number
  iternum <<- 1

  #select data we want to work on
  countries <- c(15,16,23,25)
  df <- df %>%
    filter(country %in% countries) %>%
    filter(wave == 6) %>%
    select(-c("mergeid", "hhid", "coupleid"))

  #remove low quality columns
  df <- removeBadColumns(df)

  #flag bad datapoints as NAc
  df <- setNanValues(df)

  #initialise df_all and df_model
  df_all <<- data.frame(df)
  df_model <<- data.frame(df) %>%
    select(bmi)

  rm(df)

}

#make logging function
logger <- function(string){
  sink("output.txt", append=TRUE)
  cat("\n")
  cat(string)
  sink()
}

#checks if 2 vectors can be correlated against each other
checkVectorsUseable <- function(vec1, vec2){
  nan_index_values <- c()

  for(i in 1:length(vec1)){
    if(is.na(vec1[i]) || is.na(vec2[i])){
      nan_index_values <- append(nan_index_values, i)
    }
  }

  if(length(nan_index_values) == 0){
    return(TRUE)
  }

  vec1 <- vec1[-c(nan_index_values)]
  vec2 <- vec2[-c(nan_index_values)]

  if(length(unique(vec1)) == 1 || length(unique(vec2)) == 1){
    return(FALSE)
  }else{
    return(TRUE)
  }
}


findMaxCorr <- function(df_all, df_model, iternum){
  #find maximum correlation against bmi in df_all (excluding bmi against itself)

  #log status
  logger(paste("Started iteration: ", iternum))

  #get column names and make constant in current scope
  column_names <- colnames(df_all)
  lockBinding("column_names", environment())

  #initialize empty vector for storing correlations
  correlations <- c()

  #populate correlations vector
  for(i in 1:dim(df_all)[2]){
    if(checkVectorsUseable(df_all$bmi, df_all[[i]]) == TRUE){
      correlation <- cor(df_all$bmi, df_all[[i]], use="complete.obs")
      correlations <- append(correlations, correlation)
    }else{
      correlations <- append(correlations, NA)
    }
  }

  #sort correlations vector to get largest values
  sorted_correlations <- sort(abs(correlations), decreasing=TRUE, index.return=TRUE)

  #log best (3) correlated variables in dataframe
  best_3_cors_column_names <- column_names[sorted_correlations$ix[2:4]]
  best_3_cors_values <- sorted_correlations$x[2:4]
  logger(paste("Best columns: ", best_3_cors_column_names))
  logger(paste("With absolute correlation values: ", best_3_cors_values))

  sorted_correlations$colnames <- column_names[sorted_correlations$ix]
  logger("found best correlations")
  return(sorted_correlations)
}
```

```r
#prints the most correlated values and what column it is to assist manual decision making
printNBestCorrelations <- function(correlations, n){
  correlation_vals <- correlations$x
  correlations_indexes <- correlations$ix
  correlation_names <- correlations$colnames
  for(i in 1:n){
    print(paste(correlation_names[i], correlation_vals[i], correlations_indexes[i]))
  }
  logger(paste("Printed best ", n, " correlations"))
}


switchColumnToModel <- function(df_all, df_model, column_to_switch){
  #switches column from df_all to df_model
  #note no direct return type, only modifies objects in the global scope
  df_model[[column_to_switch]] <<- df_all[[column_to_switch]]
  df_all <<- df_all %>%
    select(-c(column_to_switch))
  logger(paste("Swiched column ", column_to_switch, " from df_all to df_model"))
}


replaceNanVals <- function(df){
  #replaces the nan values in a column with the mean of the column
  df <- df %>%
    mutate_all(~ifelse(is.na(.x), mean(.x, na.rm = TRUE), .x))
  return(df)
}

removeNanVals <- function(df){
  #removes all nan values from dataframe
  df <- df %>%
    drop_na()
  return(df)
}

makeModel <- function(df_model, nan_method = "remove"){
  #remove all na from dataframe or replace with mean?
  if(nan_method == "remove"){
    df <- removeNanVals(df_model)
  }else{
    df <- replaceNanVals(df_model)
  }

  print(paste("num useable rows: ", dim(df)[1]))

  #construct linear model of bmi against rest of dataframe
  model <- lm(bmi~ ., data=df)

  #assign global
  model <<- model
  logger("Sucessfully made model")
  #no return statement necessary
}


evaluateModel <- function(){
  #model has already been made and is in global scope
  #therefore we only need to return summary and append eval to evals vector
  model_summary <- summary(model)
  r_squared <- model_summary$r.squared
  evaluations <<- append(evaluations, r_squared)

  logger("Evaluted model")
  return(model_summary)
}

#check how much the r squared has improved on this iteration of the model
quantifyImprovements <- function(){
  #improvement = r^2(n) - r^2(n-1)
  improvement <- evaluations[length(evaluations)] - evaluations[length(evaluations)-1]
  logger("Evaluated model improvement")
  logger(paste("Finished iteration", iternum))

  #add 1 to iteration number
  iternum <- iternum + 1

  return(improvement)
}
```

Appendix C

List of reasons for missing data - 3: "implausible value/suspected wrong"

- 7: "not yet coded"
- 9: "not applicable filtered"
- 10: "SHARELIFE interview" (only in wave 7)
- 11: "regular interview" (only in wave 7)
- 12: "don't know / refusal"
- 13: "not asked in this wave"
- 14: "not asked in this country"
- 15: "no information"
- 16: "no drop-off (information in drop-off in this wave)"