Tackling Obesity — Prioritizing the Risk Factors by Demographic

CSML1000 Final Project, by Group 8

Tamer Hanna tamerh@my.yorku.ca (mailto:tamerh@my.yorku.ca) Pete Gray ptgray@my.yorku.ca (mailto:ptgray@my.yorku.ca) Xiaohai Lu yu271637@my.yorku.ca (mailto:yu271637@my.yorku.ca) Haofeng Zhou zhf85@my.yorku.ca (mailto:zhf85@my.yorku.ca)

OVERVIEW

Obesity is a big problem in North American society, and contributes to increased rates of diabetes, heart disease, stroke, depression, suicide, disability, and death. All of these complications reduce quality of life and cause additional burden on the health care system.

Using this dataset, we can identify groups at highest risk based on age, gender, race, education, income, etc. Using unsupervised learning, we can look for unexpected combinations of features that lead to an increased rate of obesity. Using decision trees, we can divine the most critical factors correlated with high rates of obesity. We can do this in general, or within those clusters. Using predictive modeling, we can predict obesity rates within demographics using real, or hypothetically modified values to represent predicted improvements in obesity rates in response to hypothetical lifestyle improvements within demographics.

Within the data are a wide array of factors known to be associated with increased incidence of obesity. Some of these factors, such as ethnic background, do have predictive value with regard to obesity rates. They are, however, not factors that can be addressed. There are also factors such as eating and exercise, which are very strongly correlated with obesity and can to some small degree be addressed though public awareness campaigns and other marketing-like activities. And there are other factors, such as education and income, which are both correlated and can be addressed, but only in the very long term.

We wish, then, to explore which addressable factors are most significant, for any demographic defined by factors that are more inherent.

Business Objectives

For any given demographic, we would like to be able to predict the impact of changing different factors on that segment’s obesity rate.

This knowledge would be of value to any organizations with marketing channels into specific demographic groups, who wish to promote the kind lifestyle modifications most likely to reduce obesity rates for that group. Does focusing on eating or exercise improve obesity rates more in poorly educated young men? In well-educated middle-aged women? In senior citizens? How would getting a better education in the first place compare to difficult lifestyle modifications now?

As we explore the data, and look for patterns, and learn how to make predictions based on modifying individual features within any given group, we can find answers to these questions and more. We would then work to communicate those findings to people outside the field of Data Science, who could then harness these discoveries into the optimization of a wide variety of targeted public health initiatives.

Load Libraries

```
library(dplyr);
library(ggplot2);
library(knitr);
library(validate);
library(tidyverse);  # data manipulation
library(cluster);    # clustering algorithms
library(clusterSim);
library(factoextra);
library(fpc);
library(flexclust);
library(caTools);
library(randomForest);
```

Collect Initial Data

Load the data from the local filesystem:

```
#load data
df <- read_csv("dataset.csv");
df_origin <- df
```

Output a quick and dirty summary of the dataset, to ensure that we haven’t loaded something scrambled, or the wrong thing:

```
str(df);
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 53392 obs. of  33 variables:
##  $ YearStart                 : num  2011 2011 2011 2011 2011 ...
##  $ YearEnd                   : num  2011 2011 2011 2011 2011 ...
##  $ LocationAbbr              : chr  "AL" "AL" "AL" "AL" ...
##  $ LocationDesc              : chr  "Alabama" "Alabama" "Alabama" "Alabama" ...
##  $ Datasource                : chr  "Behavioral Risk Factor Surveillance System" "Behavioral Risk Factor Surveillance Sys
tem" "Behavioral Risk Factor Surveillance System" "Behavioral Risk Factor Surveillance System" ...
##  $ Class                     : chr  "Obesity / Weight Status" "Obesity / Weight Status" "Obesity / Weight Status" "Obesit
y / Weight Status" ...
##  $ Topic                     : chr  "Obesity / Weight Status" "Obesity / Weight Status" "Obesity / Weight Status" "Obesit
y / Weight Status" ...
##  $ Question                  : chr  "Percent of adults aged 18 years and older who have obesity" "Percent of adults aged
18 years and older who have obesity" "Percent of adults aged 18 years and older who have obesity" "Percent of adults aged 18
years and older who have obesity" ...
##  $ Data_Value_Unit           : logi  NA NA NA NA NA NA ...
##  $ Data_Value_Type           : chr  "Value" "Value" "Value" "Value" ...
##  $ Data_Value                : num  32 32.3 31.8 33.6 32.8 33.8 26.4 16.3 35.2 35.5 ...
##  $ Data_Value_Alt            : num  32 32.3 31.8 33.6 32.8 33.8 26.4 16.3 35.2 35.5 ...
##  $ Data_Value_Footnote_Symbol: chr  NA NA NA NA ...
##  $ Data_Value_Footnote       : chr  NA NA NA NA ...
##  $ Low_Confidence_Limit      : num  30.5 29.9 30 29.9 30.2 31 23.7 12.6 30.7 31.6 ...
##  $ High_Confidence_Limit     : num  33.5 34.7 33.6 37.6 35.6 36.8 29.3 20.9 40 39.6 ...
##  $ Sample_Size               : num  7304 2581 4723 1153 2402 ...
##  $ Total                     : chr  "Total" NA NA NA ...
##  $ Age(years)                : chr  NA NA NA NA ...
##  $ Education                 : chr  NA NA NA "Less than high school" ...
##  $ Gender                    : chr  NA "Male" "Female" NA ...
##  $ Income                    : chr  NA NA NA NA ...
##  $ Race/Ethnicity            : chr  NA NA NA NA ...
##  $ GeoLocation               : chr  "(32.84057112200048, -86.63186076199969)" "(32.84057112200048, -86.63186076199969)" "
(32.84057112200048, -86.63186076199969)" "(32.84057112200048, -86.63186076199969)" ...
##  $ ClassID                   : chr  "OWS" "OWS" "OWS" "OWS" ...
##  $ TopicID                   : chr  "OWS1" "OWS1" "OWS1" "OWS1" ...
##  $ QuestionID                : chr  "Q036" "Q036" "Q036" "Q036" ...
##  $ DataValueTypeID           : chr  "VALUE" "VALUE" "VALUE" "VALUE" ...
##  $ LocationID                : chr  "01" "01" "01" "01" ...
##  $ StratificationCategory1   : chr  "Total" "Gender" "Gender" "Education" ...
##  $ Stratification1           : chr  "Total" "Male" "Female" "Less than high school" ...
##  $ StratificationCategoryId1 : chr  "OVR" "GEN" "GEN" "EDU" ...
##  $ StratificationID1         : chr  "OVERALL" "MALE" "FEMALE" "EDUHS" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   YearStart = col_double(),
##   ..   YearEnd = col_double(),
##   ..   LocationAbbr = col_character(),
##   ..   LocationDesc = col_character(),
##   ..   Datasource = col_character(),
##   ..   Class = col_character(),
##   ..   Topic = col_character(),
##   ..   Question = col_character(),
##   ..   Data_Value_Unit = col_logical(),
##   ..   Data_Value_Type = col_character(),
##   ..   Data_Value = col_double(),
##   ..   Data_Value_Alt = col_double(),
##   ..   Data_Value_Footnote_Symbol = col_character(),
##   ..   Data_Value_Footnote = col_character(),
##   ..   Low_Confidence_Limit = col_double(),
##   ..   High_Confidence_Limit = col_double(),
##   ..   Sample_Size = col_double(),
##   ..   Total = col_character(),
##   ..   `Age(years)` = col_character(),
##   ..   Education = col_character(),
##   ..   Gender = col_character(),
##   ..   Income = col_character(),
##   ..   `Race/Ethnicity` = col_character(),
##   ..   GeoLocation = col_character(),
##   ..   ClassID = col_character(),
##   ..   TopicID = col_character(),
##   ..   QuestionID = col_character(),
##   ..   DataValueTypeID = col_character(),
##   ..   LocationID = col_character(),
##   ..   StratificationCategory1 = col_character(),
##   ..   Stratification1 = col_character(),
##   ..   StratificationCategoryId1 = col_character(),
##   ..   StratificationID1 = col_character()
##   .. )
```

```
head(df);
```

```
## # A tibble: 6 x 33
##   YearStart YearEnd LocationAbbr LocationDesc Datasource Class Topic
##       <dbl>   <dbl> <chr>        <chr>        <chr>      <chr> <chr>
## 1      2011    2011 AL           Alabama      Behaviora… Obes… Obes…
## 2      2011    2011 AL           Alabama      Behaviora… Obes… Obes…
## 3      2011    2011 AL           Alabama      Behaviora… Obes… Obes…
## 4      2011    2011 AL           Alabama      Behaviora… Obes… Obes…
## 5      2011    2011 AL           Alabama      Behaviora… Obes… Obes…
## 6      2011    2011 AL           Alabama      Behaviora… Obes… Obes…
## # … with 26 more variables: Question <chr>, Data_Value_Unit <lgl>,
## #   Data_Value_Type <chr>, Data_Value <dbl>, Data_Value_Alt <dbl>,
## #   Data_Value_Footnote_Symbol <chr>, Data_Value_Footnote <chr>,
## #   Low_Confidence_Limit <dbl>, High_Confidence_Limit <dbl>,
## #   Sample_Size <dbl>, Total <chr>, `Age(years)` <chr>, Education <chr>,
## #   Gender <chr>, Income <chr>, `Race/Ethnicity` <chr>, GeoLocation <chr>,
## #   ClassID <chr>, TopicID <chr>, QuestionID <chr>, DataValueTypeID <chr>,
## #   LocationID <chr>, StratificationCategory1 <chr>,
## #   Stratification1 <chr>, StratificationCategoryId1 <chr>,
## #   StratificationID1 <chr>
```

Describe Data

The data describe 9 different questions and each one has a corresponding precentage which is Data_Value column

1. Percent of adults aged 18 years and older who have an overweight classification
2. Percent of adults aged 18 years and older who have obesity
3. Percent of adults who achieve at least 150 minutes a week of moderate-intensity aerobic physical activity or 75 minutes a week of vigorous-intensity aerobic activity (or a…
4. Percent of adults who achieve at least 150 minutes a week of moderate-intensity aerobic physical activity or 75 minutes a week of vigorous-intensity aerobic physical activ…
5. Percent of adults who achieve at least 300 minutes a week of moderate-intensity aerobic physical activity or 150 minutes a week of vigorous-intensity aerobic activity (or …
6. Percent of adults who engage in muscle-strengthening activities on 2 or more days a week 7. Percent of adults who engage in no leisure-time physical activity 8. Percent of adults who report consuming fruit less than one time daily 9. Percent of adults who report consuming vegetables less than one time daily

For each of there questions, there are 6 different categories

1. Age (years)
2. Education
3. Gender
4. Income
5. Race/Ethnicity
6. Total

For each of there categories, which has different values

1. Age (years) (6 levels)

- 18 - 24
- 25 - 34
- 35 - 44
- 45 - 54
- 55 - 64
- 65 or older

2. Education (4 levels)

- Less than high school
- High school graduate
- Some college or technical school

- College graduate

3. Gender (2 levels)

- Male
- Female

4. Income (7 levels)

- Less than $15,000
- $15,000 - $24,999
- $25,000 - $34,999
- $35,000 - $49,999
- $50,000 - $74,999
- $75,000 or greater
- Data not reported

5. Race/Ethnicity (8 levels)

- 2 or more races
- American Indian/Alaska Native
- Asian
- Hawaiian/Pacific Islander
- Hispanic
- Non-Hispanic Black
- Non-Hispanic White
- Other

6. Total (1 levels)

- Total

Data cleaning

At first glance, some columns are just described another. Let’s look at one by one from left to right. And select the necessary columns. Firs of all, converting all character to factor

```
df$LocationAbbr <- as.factor(df$LocationAbbr)
df$LocationDesc <- as.factor(df$LocationDesc)
df$Datasource <- as.factor(df$Datasource)
df$Class <- as.factor(df$Class)
df$Topic <- as.factor(df$Topic)
df$Data_Value_Unit <- as.factor(df$Data_Value_Unit)
df$Data_Value_Type <- as.factor(df$Data_Value_Type)
df$Data_Value_Footnote_Symbol <- as.factor(df$Data_Value_Footnote_Symbol)
df$Data_Value_Footnote <- as.factor(df$Data_Value_Footnote)
df$Total <- as.factor(df$Total)
df$Education <- as.factor(df$Education)
df$Gender <- as.factor(df$Gender)
df$Income <- as.factor(df$Income)
df$ClassID <- as.factor(df$ClassID)
df$TopicID <- as.factor(df$TopicID)
df$QuestionID <- as.factor(df$QuestionID)
df$DataValueTypeID <- as.factor(df$DataValueTypeID)
df$LocationID <- as.factor(df$LocationID)
df$DataValueTypeID <- as.factor(df$DataValueTypeID)
df$StratificationCategory1 <- as.factor(df$StratificationCategory1)
df$Stratification1 <- as.factor(df$Stratification1)
df$StratificationCategoryId1 <- as.factor(df$StratificationCategoryId1)
df$StratificationID1 <- as.factor(df$StratificationID1)
```

Next, walk through each column.

YearStart vs YearEnd

Group by YearStart and YearEnd, if they are same, we can keep one and remove the other.

```
df %>% group_by(YearStart,YearEnd) %>% summarise(count=n())
```

```
## # A tibble: 6 x 3
## # Groups:   YearStart [6]
##   YearStart YearEnd count
##       <dbl>   <dbl> <int>
## 1      2011    2011 13104
## 2      2012    2012  4368
## 3      2013    2013 13160
## 4      2014    2014  4536
## 5      2015    2015 13604
## 6      2016    2016  4620
```

We can see YearStart and YearEnd are always the same. let’s remove YearEnd. And we also found each year contains different size of data. 2015 > 2013 > 2011 > 2016 > 2014 > 2012

```
df$YearEnd <- NULL
```

Datasource

Datasource has only one level. We can remove it as well.

```
df$Datasource <- NULL
```

Class vs Topic, ClassID vs TopicID

Class and Topic have 3 levels. Let’s take a look whether those 3 level are same or not.

```
levels(df$Class)
```

```
## [1] "Fruits and Vegetables"   "Obesity / Weight Status"
## [3] "Physical Activity"
```

```
levels(df$Topic)
```

```
## [1] "Fruits and Vegetables - Behavior" "Obesity / Weight Status"
## [3] "Physical Activity - Behavior"
```

Class levels : Fruits and Vegetables, Obesity / Weight Status, Physical Activity Topic levels: Fruits and Vegetables - Behavior, Obesity / Weight Status, Physical Activity - Behavior They are identical, so that we can remove one, which is Topic

```
df$Topic <- NULL
```

ClassID and TopicID columms are just abbr/transform value of Class and Topic colums. We can continous Class column and ignore there two.

```
df$ClassID <- NULL
df$TopicID <- NULL
```

Question vs QuestionID

Again, QuestionID is a value tranfromation of Question column. We keep Question and remove Question.

```
df %>% group_by(QuestionID, Question) %>% summarise()
```

```
## # A tibble: 9 x 2
## # Groups:   QuestionID [9]
##   QuestionID Question
##   <fct>      <chr>
## 1 Q018       Percent of adults who report consuming fruit less than one ti…
## 2 Q019       Percent of adults who report consuming vegetables less than o…
## 3 Q036       Percent of adults aged 18 years and older who have obesity
## 4 Q037       Percent of adults aged 18 years and older who have an overwei…
## 5 Q043       Percent of adults who achieve at least 150 minutes a week of …
## 6 Q044       Percent of adults who achieve at least 150 minutes a week of …
## 7 Q045       Percent of adults who achieve at least 300 minutes a week of …
## 8 Q046       Percent of adults who engage in muscle-strengthening activiti…
## 9 Q047       Percent of adults who engage in no leisure-time physical acti…
```

```
df$Question <- NULL
```

There are total 9 levels, which means this dataset might be able to split by 9 different group. Below is the mapping: Q018 - Percent of adults who report consuming fruit less than one time daily Q019 - Percent of adults who report consuming vegetables less than one time daily Q036 - Percent of adults aged 18 years and older who have obesity Q037 - Percent of adults aged 18 years and older who have an overweight classification Q043 - Percent of adults who achieve at least 150 minutes a week of moderate-intensity aerobic physical activity or 75 minutes a week of vigorous-intensity aerobic activity (or an equivalent combination) Q044 - Percent of adults who achieve at least 150 minutes a week of moderate-intensity aerobic physical activity or 75 minutes a week of vigorous-intensity aerobic physical activity and engage in muscle-strengthening activities on 2 or more days a week Q045 - Percent of adults who achieve at least 300 minutes a week of moderate-intensity aerobic physical activity or 150 minutes a week of vigorous-intensity aerobic activity (or an equivalent combination) Q046 - Percent of adults who engage in muscle-strengthening activities on 2 or more days a week Q047 - Percent of adults who engage in no leisure-time physical activity

According to each question, it is obviously Q036 and Q037 are relative to weight status/obesity problem. Q018 and Q019 are relative to healthy diet, fruit and vegetable consuming. Q043, Q044, Q045, Q046 and Q047 are relative to physical activity.

For this final project, we will foucus on obesity rate, which is relative to questions Q036 and Q037.

Data_Value_Unit vs Data_Value_Type vs DataValueTypeID

Since Data_Value_Unit is all NA. And Data_Value_Type and DataValueTypeID are all “VALUE”. We can remove them.

```
df$Data_Value_Unit <- NULL
df$Data_Value_Type <- NULL
df$DataValueTypeID <- NULL
```

Data_Value vs Data_Value_Alt

Based on first glance, there 2 columns should be identical. Let’s use group by to confirm.

```
df %>% group_by(Data_Value,Data_Value_Alt)
```

```
## # A tibble: 53,392 x 24
## # Groups:   Data_Value, Data_Value_Alt [670]
##    YearStart LocationAbbr LocationDesc Class Data_Value Data_Value_Alt
##        <dbl> <fct>        <fct>        <fct>      <dbl>          <dbl>
##  1      2011 AL           Alabama      Obes…       32             32  
##  2      2011 AL           Alabama      Obes…       32.3           32.3
##  3      2011 AL           Alabama      Obes…       31.8           31.8
##  4      2011 AL           Alabama      Obes…       33.6           33.6
##  5      2011 AL           Alabama      Obes…       32.8           32.8
##  6      2011 AL           Alabama      Obes…       33.8           33.8
##  7      2011 AL           Alabama      Obes…       26.4           26.4
##  8      2011 AL           Alabama      Obes…       16.3           16.3
##  9      2011 AL           Alabama      Obes…       35.2           35.2
## 10      2011 AL           Alabama      Obes…       35.5           35.5
## # … with 53,382 more rows, and 18 more variables:
## #   Data_Value_Footnote_Symbol <fct>, Data_Value_Footnote <fct>,
## #   Low_Confidence_Limit <dbl>, High_Confidence_Limit <dbl>,
## #   Sample_Size <dbl>, Total <fct>, `Age(years)` <chr>, Education <fct>,
## #   Gender <fct>, Income <fct>, `Race/Ethnicity` <chr>, GeoLocation <chr>,
## #   QuestionID <fct>, LocationID <fct>, StratificationCategory1 <fct>,
## #   Stratification1 <fct>, StratificationCategoryId1 <fct>,
## #   StratificationID1 <fct>
```

After group by, total rows is 53,392, which is same as original dataset. So that, they are the same. Let’s remove Data_Value_Alt.

```
df$Data_Value_Alt <- NULL
```

Data_Value vs Low_Confidence_Limit vs High_Confidence_Limit

High_Confidence_Limit and Low_Confidence_Limit. These two columns are not very clear what is actually representing. Based the on column name, it looks like the min and max value of the Data_Value in each group. So that, we can remove two columns for now. And we would reley on Data_Value more than those High_Confidence_Limit and Low_Confidence_Limit

```
df$Low_Confidence_Limit <- NULL;
df$High_Confidence_Limit <- NULL;
```

Data_Value_Footnote_Symbol vs Data_Value_Footnote

According to the name, these 2 columns are the same. when “Data_Value_Footnote_Symbol = ~” is equal “Data_Value_Footnote_Symbol = Data not available because sample size is insufficient.” Let’s filter out when Data_Value_Footnote_Symbol is ~.

```
head(df %>% filter(Data_Value_Footnote_Symbol == "~"))
```

```
## # A tibble: 6 x 21
##   YearStart LocationAbbr LocationDesc Class Data_Value Data_Value_Foot…
##       <dbl> <fct>        <fct>        <fct>      <dbl> <fct>
## 1      2011 AL           Alabama      Obes…         NA ~
## 2      2011 AL           Alabama      Obes…         NA ~
## 3      2011 AL           Alabama      Obes…         NA ~
## 4      2011 AL           Alabama      Obes…         NA ~
## 5      2011 AL           Alabama      Obes…         NA ~
## 6      2011 AL           Alabama      Obes…         NA ~
## # … with 15 more variables: Data_Value_Footnote <fct>, Sample_Size <dbl>,
## #   Total <fct>, `Age(years)` <chr>, Education <fct>, Gender <fct>,
## #   Income <fct>, `Race/Ethnicity` <chr>, GeoLocation <chr>,
## #   QuestionID <fct>, LocationID <fct>, StratificationCategory1 <fct>,
## #   Stratification1 <fct>, StratificationCategoryId1 <fct>,
## #   StratificationID1 <fct>
```

When Data_Value_Footnote_Symbol is ~, Data_Value value will be NA. We need to clean data. So that, we need to remove those records.

```
df <- df %>% filter(is.na(Data_Value_Footnote_Symbol))
```

After that, we can drop Data_Value_Footnote_Symbol and Data_Value_Footnote.

```
df$Data_Value_Footnote_Symbol <- NULL
df$Data_Value_Footnote <- NULL
```

Total, Age(years), Education, Gender, Income and Race/Ethnicity vs StratificationCategory1 and Stratification1

Let’s take a look the relationship between these columns.

```
head(df,5);
```

```
## # A tibble: 5 x 19
##   YearStart LocationAbbr LocationDesc Class Data_Value Sample_Size Total
##       <dbl> <fct>        <fct>        <fct>      <dbl>       <dbl> <fct>
## 1      2011 AL           Alabama      Obes…       32          7304 Total
## 2      2011 AL           Alabama      Obes…       32.3        2581 <NA>
## 3      2011 AL           Alabama      Obes…       31.8        4723 <NA>
## 4      2011 AL           Alabama      Obes…       33.6        1153 <NA>
## 5      2011 AL           Alabama      Obes…       32.8        2402 <NA>
## # … with 12 more variables: `Age(years)` <chr>, Education <fct>,
## #   Gender <fct>, Income <fct>, `Race/Ethnicity` <chr>, GeoLocation <chr>,
## #   QuestionID <fct>, LocationID <fct>, StratificationCategory1 <fct>,
## #   Stratification1 <fct>, StratificationCategoryId1 <fct>,
## #   StratificationID1 <fct>
```

According to the data, it is easy to find out columns “Total, Age(years), Education, Gender, Income and Race/Ethnicity”, only one column has value. For example, if column “Total” has value, columns “Education, Gender, Income and Race/Ethnicity”’s value are NA. And StratificationCategory1’s value is “Total”. If column “Gender” has value, columns “Education, Total, Income and Race/Ethnicity”’s value are NA. And StratificationCategory1’s value is “Gender”. Column Stratification1’s value is “MALE” or “FEMALE”. Thereforce, StratificationCategory1 is summary of columns “Total, Age(years), Education, Gender, Income and Race/Ethnicity”. And Stratification1 colmn is the actual value. In this case, we can reply on “StratificationCategory1 and Stratification1”. And remove “Total, Age(years), Education, Gender, Income and Race/Ethnicity”

```
df$Total <- NULL
df$`Age(years)` <- NULL
df$Education <- NULL
df$Gender <- NULL
df$Income <- NULL
df$`Race/Ethnicity` <- NULL
```

Let’s group by StratificationCategory1 and Stratification1 and see how many different combinations in the dataset.

```
df %>% group_by(StratificationCategory1) %>% summarise()
```

```
## # A tibble: 6 x 1
##   StratificationCategory1
##   <fct>
## 1 Age (years)
## 2 Education
## 3 Gender
## 4 Income
## 5 Race/Ethnicity
## 6 Total
```

```
df %>% group_by(StratificationCategory1) %>% summarise(CategoryDistinctValue = n_distinct(Stratification1))
```

```
## # A tibble: 6 x 2
##   StratificationCategory1 CategoryDistinctValue
##   <fct>                                   <int>
## 1 Age (years)                                 6
## 2 Education                                   4
## 3 Gender                                      2
## 4 Income                                      7
## 5 Race/Ethnicity                              8
## 6 Total                                       1
```

We can see there are 6 categories. And each category might contain more than value. For example, in “Age(years)” category, it has 6 different values.

StratificationCategoryId1 vs StratificationCategoryId1

StratificationCategoryId1 vs StratificationCategoryId1 are the transform value for StratificationCategory1 vs StratificationCategory1. So that, we can remove them.

```
df$StratificationCategoryId1 <- NULL
df$StratificationID1 <- NULL
```

LocationID vs LocationDesc vs LocationAbbr vs GeoLocation

Based on columns’ name, they should represent the same thing. Let’s group by them and see whether we can remove the redundant columns or not.

```
df %>% group_by(LocationID, LocationAbbr, GeoLocation, LocationDesc) %>% summarise()
```

```
## # A tibble: 55 x 4
## # Groups:   LocationID, LocationAbbr, GeoLocation [55]
##    LocationID LocationAbbr GeoLocation                    LocationDesc
##    <fct>      <fct>        <chr>                          <fct>
##  1 01         AL           (32.84057112200048, -86.63186… Alabama
##  2 02         AK           (64.84507995700051, -147.7220… Alaska
##  3 04         AZ           (34.865970280000454, -111.763… Arizona
##  4 05         AR           (34.74865012400045, -92.27449… Arkansas
##  5 06         CA           (37.63864012300047, -120.9999… California
##  6 08         CO           (38.843840757000464, -106.133… Colorado
##  7 09         CT           (41.56266102000046, -72.64984… Connecticut
##  8 10         DE           (39.008830667000495, -75.5777… Delaware
##  9 11         DC           (38.89037138500049, -77.03196… District of Colu…
## 10 12         FL           (28.932040377000476, -81.9289… Florida
## # … with 45 more rows
```

After group by, we can clearly to see there are identical and repreesnt in different way. We can keep LocationAbbr and remove others.

```
df$LocationDesc <- NULL
df$GeoLocation <- NULL
df$LocationID <- NULL
```

Some of Location are not including in US’s 50+ states or some data is crossing the whole US. We need to remove those as well.

```
removeLocation <- c("US","GU","PR","VI");
df <- df %>% filter(!LocationAbbr %in% removeLocation);
```

Let’s take a look LocationAbbr column

```
summary(df$LocationAbbr)
```

```
##  AK  AL  AR  AZ  CA  CO  CT  DC  DE  FL  GA  GU  HI  IA  ID  IL  IN  KS 
## 920 889 897 948 963 963 939 908 891 948 939   0 930 861 864 900 917 954 
##  KY  LA  MA  MD  ME  MI  MN  MO  MS  MT  NC  ND  NE  NH  NJ  NM  NV  NY 
## 911 887 953 962 873 945 953 892 819 876 949 833 948 864 951 942 932 920 
##  OH  OK  OR  PA  PR  RI  SC  SD  TN  TX  US  UT  VA  VI  VT  WA  WI  WV 
## 952 927 879 903   0 912 946 864 850 934   0 936 939   0 854 984 904 850 
##  WY 
## 862
```

However, the filtered values are still there. We need to remove them from factor level.

```
df[] <- lapply(df, function(x) if(is.factor(x)) factor(x) else x)
```

Data Cleaning Round 1 - Summary

Let’s check the dataset again and see what does the dataset looks like after first round cleaning

```
str(df)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 46537 obs. of  8 variables:
##  $ YearStart              : num  2011 2011 2011 2011 2011 ...
##  $ LocationAbbr           : Factor w/ 51 levels "AK","AL","AR",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Class                  : Factor w/ 3 levels "Fruits and Vegetables",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Data_Value             : num  32 32.3 31.8 33.6 32.8 33.8 26.4 16.3 35.2 35.5 ...
##  $ Sample_Size            : num  7304 2581 4723 1153 2402 ...
##  $ QuestionID             : Factor w/ 9 levels "Q018","Q019",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ StratificationCategory1: Factor w/ 6 levels "Age (years)",..: 6 3 3 2 2 2 2 1 1 1 ...
##  $ Stratification1        : Factor w/ 28 levels "$15,000 - $24,999",..: 28 23 17 22 19 27 15 6 8 9 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   YearStart = col_double(),
##   ..   YearEnd = col_double(),
##   ..   LocationAbbr = col_character(),
##   ..   LocationDesc = col_character(),
##   ..   Datasource = col_character(),
##   ..   Class = col_character(),
##   ..   Topic = col_character(),
##   ..   Question = col_character(),
##   ..   Data_Value_Unit = col_logical(),
##   ..   Data_Value_Type = col_character(),
##   ..   Data_Value = col_double(),
##   ..   Data_Value_Alt = col_double(),
##   ..   Data_Value_Footnote_Symbol = col_character(),
##   ..   Data_Value_Footnote = col_character(),
##   ..   Low_Confidence_Limit = col_double(),
##   ..   High_Confidence_Limit = col_double(),
##   ..   Sample_Size = col_double(),
##   ..   Total = col_character(),
##   ..   `Age(years)` = col_character(),
##   ..   Education = col_character(),
##   ..   Gender = col_character(),
##   ..   Income = col_character(),
##   ..   `Race/Ethnicity` = col_character(),
##   ..   GeoLocation = col_character(),
##   ..   ClassID = col_character(),
##   ..   TopicID = col_character(),
##   ..   QuestionID = col_character(),
##   ..   DataValueTypeID = col_character(),
##   ..   LocationID = col_character(),
##   ..   StratificationCategory1 = col_character(),
##   ..   Stratification1 = col_character(),
##   ..   StratificationCategoryId1 = col_character(),
##   ..   StratificationID1 = col_character()
##   .. )
```

And let have a quick check whether any NA value in our dataset or not.

```
apply(df, 2, function(x) any(is.na(x) | is.infinite(x)))
```

```
##               YearStart             LocationAbbr                    Class 
##                   FALSE                    FALSE                    FALSE 
##              Data_Value              Sample_Size               QuestionID 
##                   FALSE                    FALSE                    FALSE 
## StratificationCategory1          Stratification1 
##                   FALSE                    FALSE
```

Great. There is no NA value found in our dataset. After dataset clean up, we have 10 columns and 48346 records(rows). Comparing to original dataset, we removed 23 columns and 5046 records(rows).

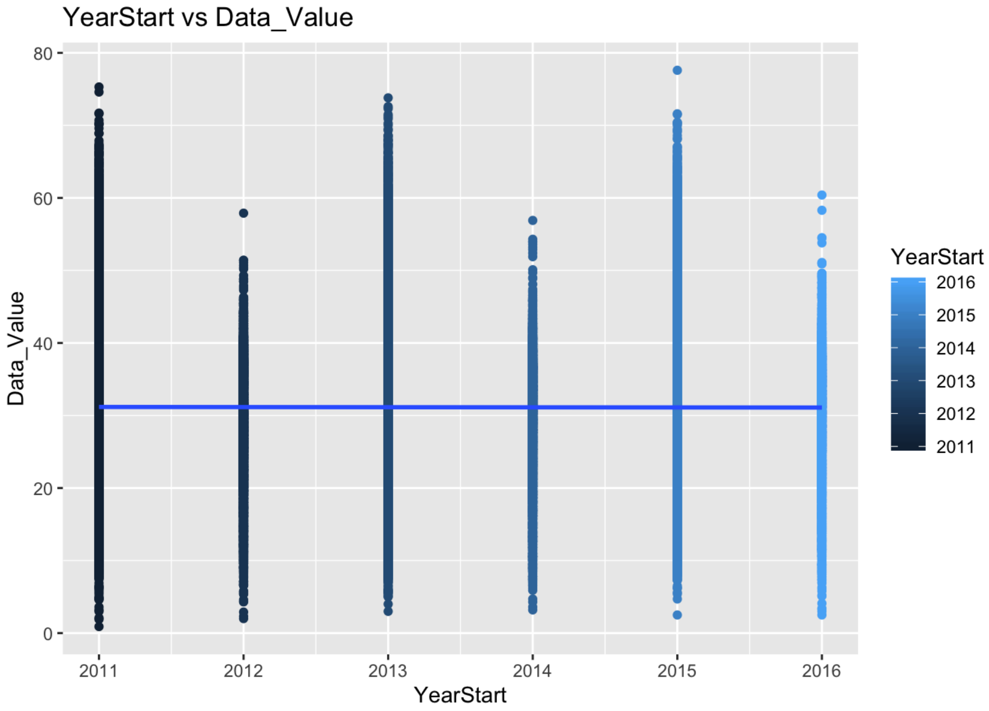
After the data clean up, let's explore data and try to find out some patterns.

Explore Data

We'll plot some basic graphs, to ensure that the data conform to our limited domain understanding.

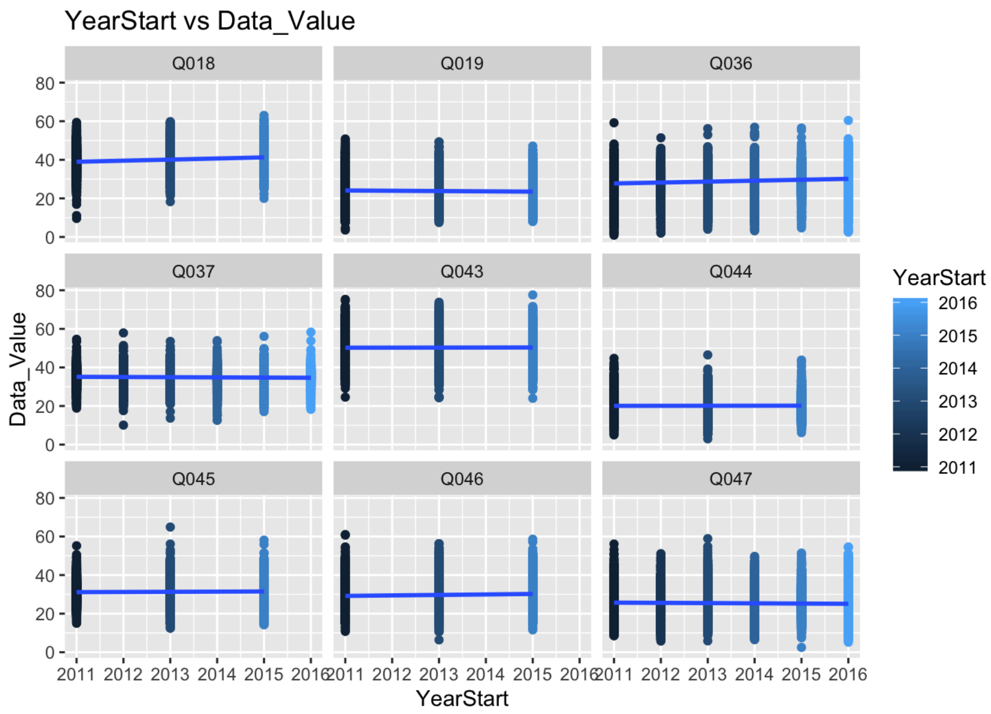
YearStart vs Data_Value - we expect there should a pattern between these 2 columns. Either increase by each year or decrease.

```
df %>% ggplot(aes(x=YearStart, y=Data_Value, color=YearStart)) +
  geom_point() +
  geom_smooth(method = lm, se = FALSE) +
  labs(title="YearStart vs Data_Value", x="YearStart", y="Data_Value")
```



However, we can not find a linear relationship. How about if we consider Quesiton in this graph. It might show a clear pattern ?

```
df %>% ggplot(aes(x=YearStart, y=Data_Value, color=YearStart)) +
  geom_point() +
  facet_wrap(~QuestionID) +
  geom_smooth(method = lm, se = FALSE) +
  labs(title="YearStart vs Data_Value", x="YearStart", y="Data_Value");
```



This graph tells us a very important information. Not all questions have a data from 2011-2016. Some of them are missing data in 2012, 2014 and 2016. Maybe we can select different questions such as Q036 and Q037 becuase they have full data from 2011-2016 and they are all about obesity problem. We could alos choose a single year, such as 2015 all questions have full data in year 2015.(We could invest this one if we have more time.) First of all, let’s choose Q036 and Q037 and take a look the relationship for each categories.

Obesity Question Analysis

- Q036 - Percent of adults aged 18 years and older who have obesity.

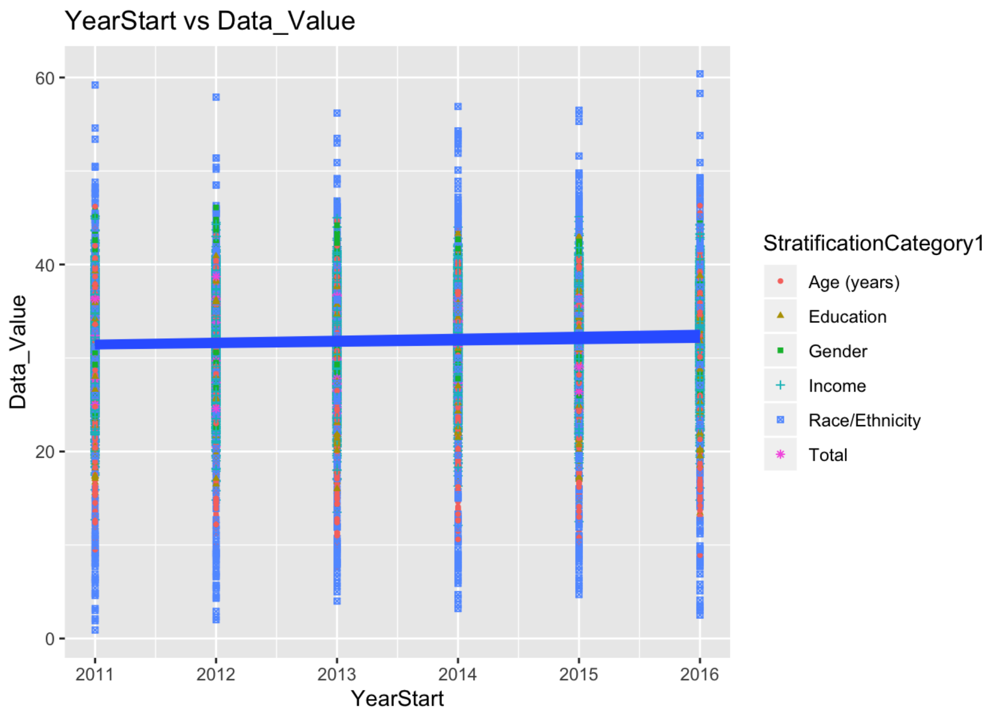
- Q037 - Percent of adults aged 18 years and older who have an overweight classification

Let’s create a new dataset and only contains data relative to Q036 and Q037.

```
q1 <- df %>% filter(QuestionID %in% c("Q036","Q037") );
```

Year vs Obesity Rate

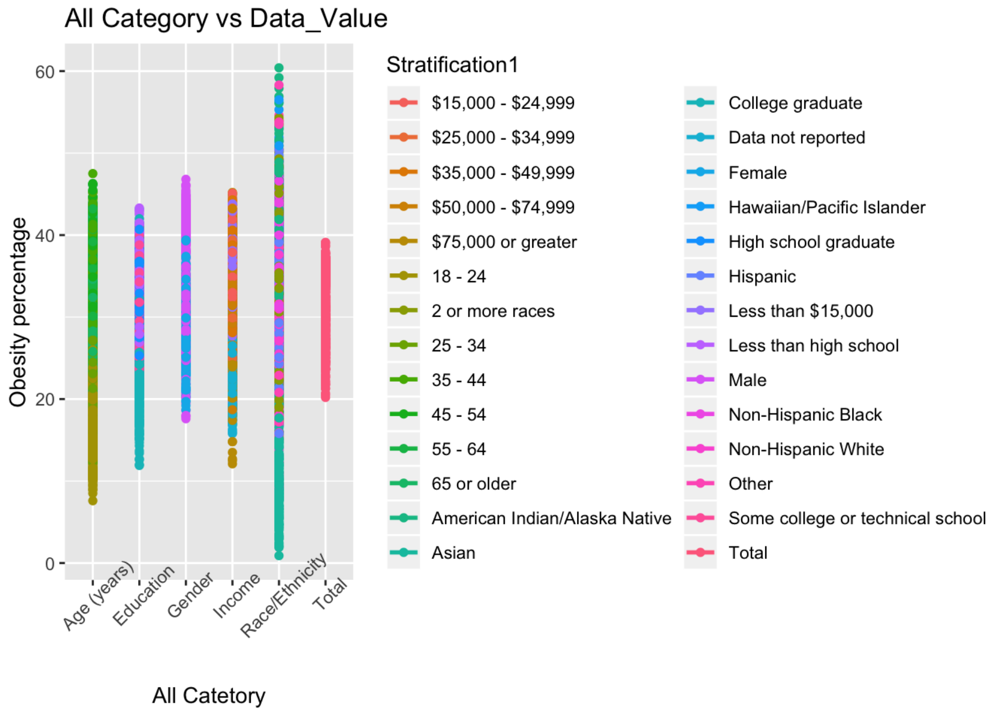
```
q1 %>% ggplot(aes(x=YearStart, y=Data_Value, shape = StratificationCategory1)) +
  geom_point(aes(colour = StratificationCategory1), size = 1) +
  geom_smooth(method = lm, se = FALSE) +
  labs(title="YearStart vs Data_Value", x="YearStart", y="Data_Value");
```



We still can see a linear pattern between year and data value. It increases each year.

Categories vs Obesity Rate

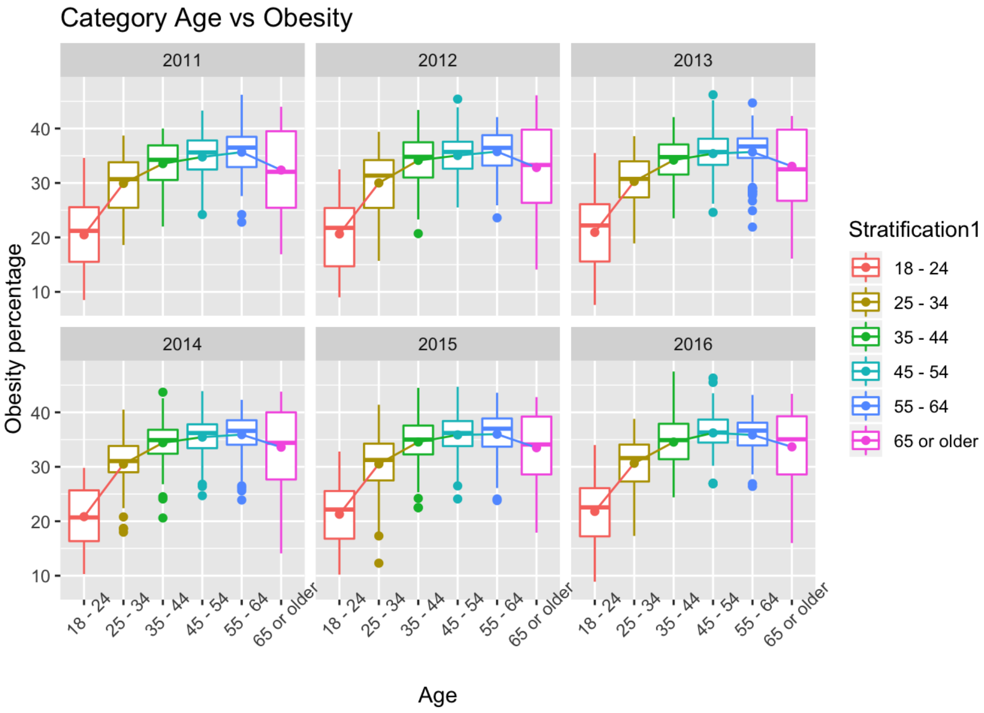
```
q1 %>% ggplot(aes(x=StratificationCategory1, y=Data_Value, color = Stratification1 ) ) +
  geom_point() +
  geom_smooth(method = lm, se = FALSE) +
  theme(axis.text.x = element_text(angle=45)) +
  labs(title="All Category vs Data_Value", x="All Catetory", y="Obesity percentage")
```



Different category has different perceatage of obesity. Race/Ethnicity category has wider range than others. Age category has a high percetage of obesity, it is all green starting from 30 in y axis. Let's take look into each category.

Category - Age (years) vs Obesity Rate

```
q1_age <- q1 %>% filter(StratificationCategory1 == "Age (years)");
q1_age %>% ggplot(aes(x=Stratification1, y=Data_Value, color=Stratification1) ) +
  geom_boxplot() +
  facet_wrap(~ YearStart) +
  theme(axis.text.x = element_text(angle=45)) +
  stat_summary(fun.y=mean, geom="line", aes(group=1))  +
  stat_summary(fun.y=mean, geom="point") +
  labs(title="Category Age vs Obesity", x="Age", y="Obesity percentage")
```



The graph is a little bit hard to find out which one has highest rate. Let's take a look the mean values.

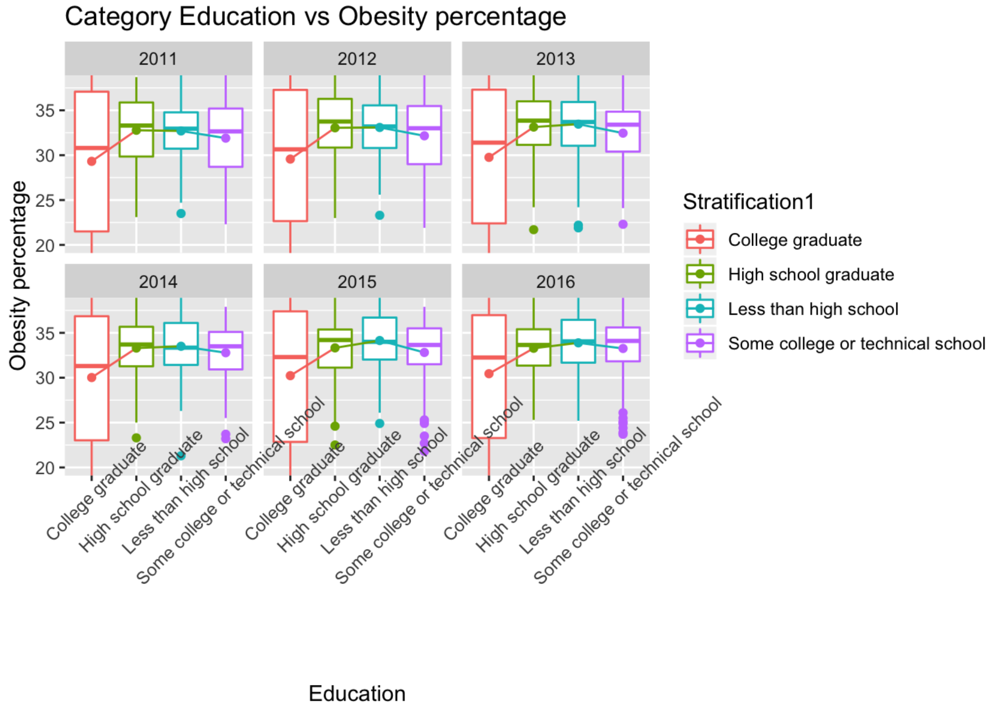
```
q1_age %>% group_by(Stratification1) %>% dplyr::summarize(Mean = mean(Data_Value, na.rm=TRUE))
```

```
## # A tibble: 6 x 2
##   Stratification1  Mean
##   <fct>           <dbl>
## 1 18 - 24          21.0
## 2 25 - 34          30.3
## 3 35 - 44          34.2
## 4 45 - 54          35.5
## 5 55 - 64          35.8
## 6 65 or older      33.2
```

It is clear obesity rate increases when age increase. Peek is around 55 - 64. And it drops after.

Category - Education vs Obesity Rate

```
q1_education <- q1 %>% filter(StratificationCategory1 == "Education");
q1_education %>% ggplot(aes(x=Stratification1, y=Data_Value,color=Stratification1)) +
  geom_boxplot(position = position_dodge2(preserve = "total")) +
  facet_wrap(~ YearStart, ncol=3) +
  theme(axis.text.x = element_text(angle=45)) +
  coord_cartesian(ylim=c(20,38)) +
  stat_summary(fun.y=mean, geom="line", aes(group=1))  +
  stat_summary(fun.y=mean, geom="point") +
  labs(title="Category Education vs Obesity percentage", x="Education", y="Obesity percentage")
```



The graph is a little bit hard to find out which one has highest rate. Let's take a look the mean values.

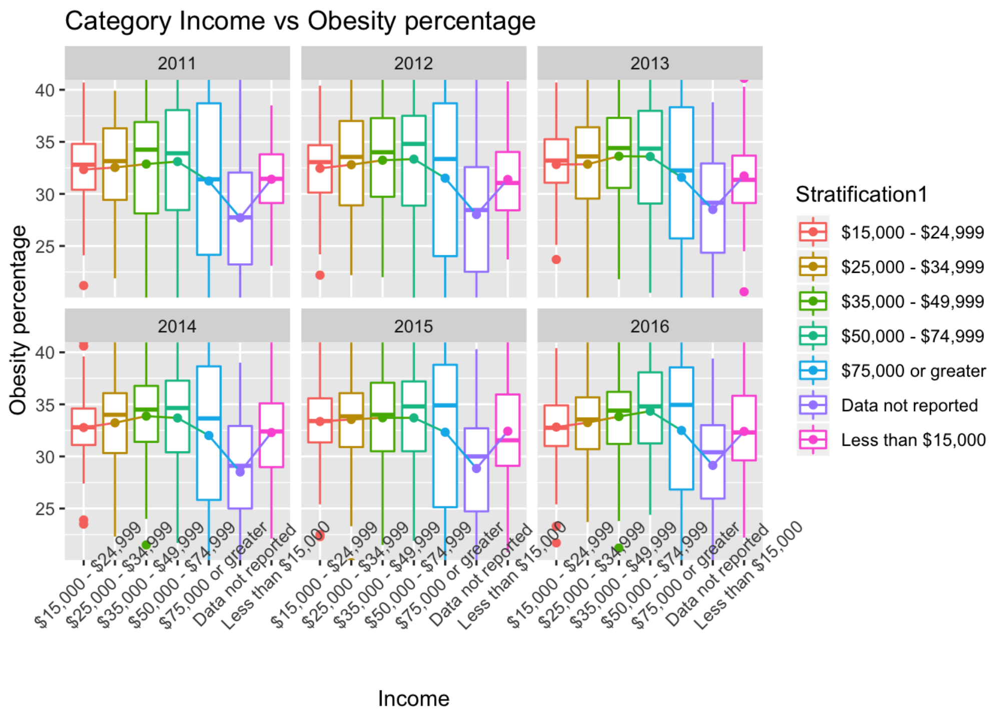
```
q1_education %>% group_by(Stratification1) %>% dplyr::summarize(Mean = mean(Data_Value, na.rm=TRUE))
```

```
## # A tibble: 4 x 2
##   Stratification1                   Mean
##   <fct>                            <dbl>
## 1 College graduate                  29.9
## 2 High school graduate              33.1
## 3 Less than high school             33.5
## 4 Some college or technical school  32.6
```

This graph tell us lower educaiton level(High school graduate & Less than high school) seems has higher obesity rate. According to the mean value, we can say peole has Less than high school education has highest obesity rate.

Category - Income vs Obesity Rate

```
q1_income <- q1 %>% filter(StratificationCategory1 == "Income");
q1_income %>% ggplot(aes(x=Stratification1, y=Data_Value, color=Stratification1) ) +
  geom_boxplot() +
  facet_wrap(~ YearStart) +
  theme(axis.text.x = element_text(angle=45)) +
  coord_cartesian(ylim=c(21,40)) +
  stat_summary(fun.y=mean, geom="line", aes(group=1))  +
  stat_summary(fun.y=mean, geom="point") +
  labs(title="Category Income vs Obesity percentage", x="Income", y="Obesity percentage")
```



The graph is a little bit hard to find out which one has highest rate. Let’s take a look the mean values.

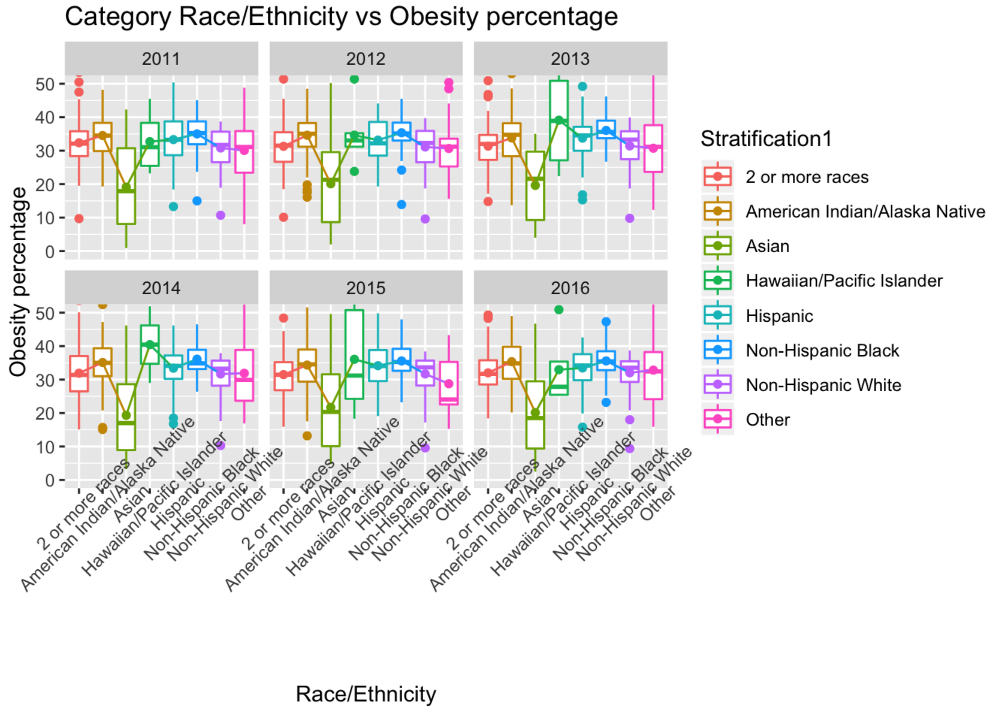
```
q1_income %>% group_by(Stratification1) %>% dplyr::summarize(Mean = mean(Data_Value, na.rm=TRUE))
```

```
## # A tibble: 7 x 2
##   Stratification1     Mean
##   <fct>              <dbl>
## 1 $15,000 - $24,999   32.8
## 2 $25,000 - $34,999   33.0
## 3 $35,000 - $49,999   33.5
## 4 $50,000 - $74,999   33.6
## 5 $75,000 or greater  31.9
## 6 Data not reported   28.5
## 7 Less than $15,000   31.9
```

Mean value tells us two boundaries case ($75,000+ and $15,000-) has lowest mean obesity rate. However, people has income between $75,000 or greater, which has bigger range than others. And the mean value is increasing year by year.

Category - Race/Ethnicity vs Obesity Rate

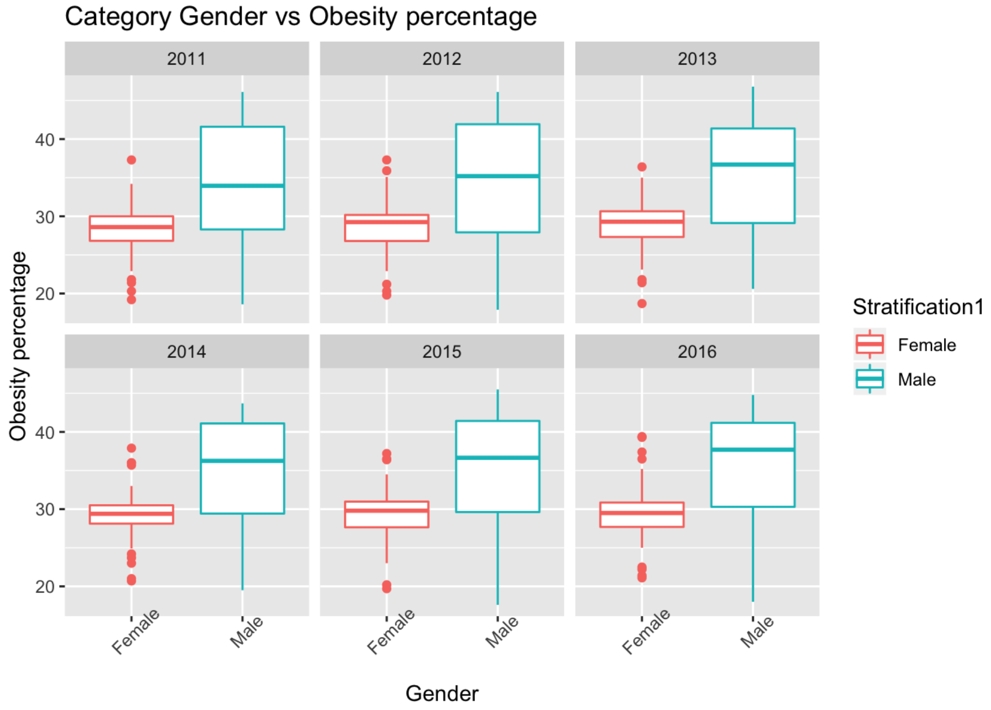
```
q1_race <- q1 %>% filter(StratificationCategory1 == "Race/Ethnicity");
q1_race %>% ggplot(aes(x=Stratification1, y=Data_Value, color=Stratification1) ) +
  geom_boxplot() +
  facet_wrap(~ YearStart) +
  coord_cartesian(ylim=c(0,50)) +
  theme(axis.text.x = element_text(angle=45)) +
  stat_summary(fun.y=mean, geom="line", aes(group=1))  +
  stat_summary(fun.y=mean, geom="point") +
  labs(title="Category Race/Ethnicity vs Obesity percentage", x="Race/Ethnicity", y="Obesity percentage")
```



It is very clear Asian people has lowest obesity rate.

Category - Gender vs Obesity Rate

```
q1_gender <- q1 %>% filter(StratificationCategory1 == "Gender");
q1_gender %>% ggplot(aes(x=Stratification1, y=Data_Value, color=Stratification1 ) ) +
  geom_boxplot() +
  facet_wrap(~ YearStart) +
  geom_smooth(method = lm, se = FALSE) +
  theme(axis.text.x = element_text(angle=45)) +
  labs(title="Category Gender vs Obesity percentage", x="Gender", y="Obesity percentage")
```



For this graph, we only can see male has wider range than femail. And male’s mean value is higher than female.

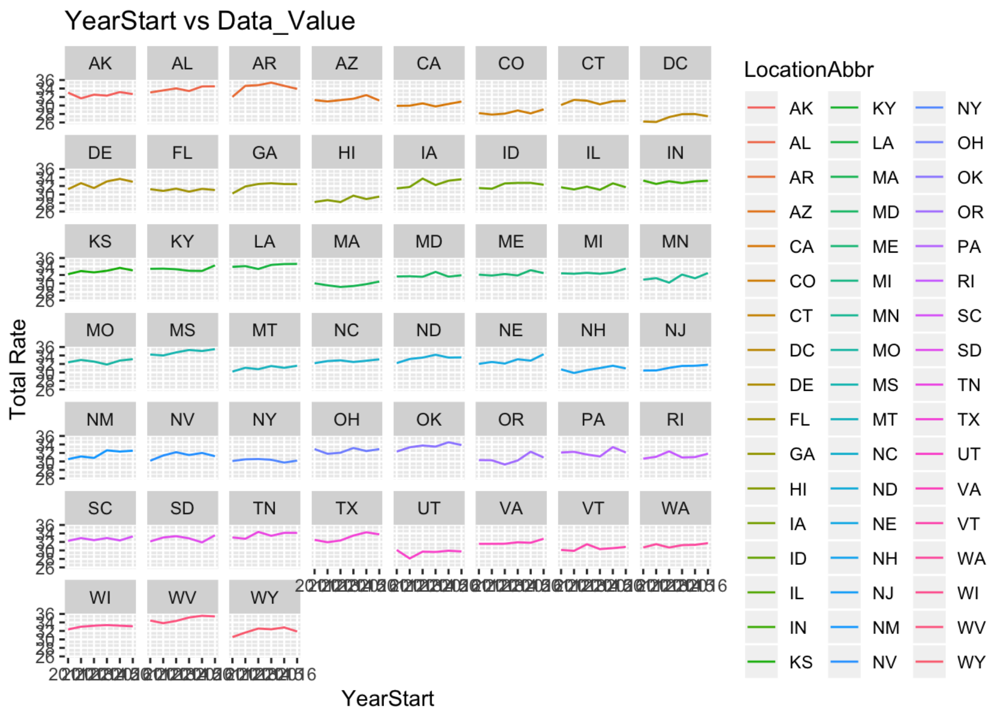
> For “Percent of adults aged 18 years and older who have obesity” question, categories Gender, Education, Income, Age and Race/Ethnicity, which has significant impact on obesity. For example, people around 55-64, whose is male, who has lower education level and income less than $15,000, they will have higher chances to have a obesity health problem.

How about the obesity rate vs location(each state)

Let’s group the dataset by year, location and categories

```
q1_groupby <- q1 %>% group_by(YearStart,LocationAbbr) %>% summarise(total = sum(Data_Value) / n() )
```

```
q1_groupby %>% ggplot(aes(x=YearStart, y=total, color = LocationAbbr)) +
  geom_line() +
  facet_wrap(~LocationAbbr) +
  labs(title="YearStart vs Data_Value", x="YearStart", y="Total Rate")
```



Corssing all state, we can find a kind of linear pattern. Rate is increasing year by year.

Unsupervised Learning - build a clustering model

Build Kmeams Model #1

Let’s build a simple clustering model. We need to convert factor to numeric

```
q1_model_df <- q1;
q1_model_df$Sample_Size <- NULL;
q1_model_df$LocationAbbr <- as.integer(q1_model_df$LocationAbbr);
q1_model_df$Class <- as.integer(q1_model_df$Class);
q1_model_df$QuestionID <- as.integer(q1_model_df$QuestionID);
q1_model_df$Stratification1 <- as.integer(q1_model_df$Stratification1);
q1_model_df$StratificationCategory1 <- as.integer(q1_model_df$StratificationCategory1)
```

Scaling data

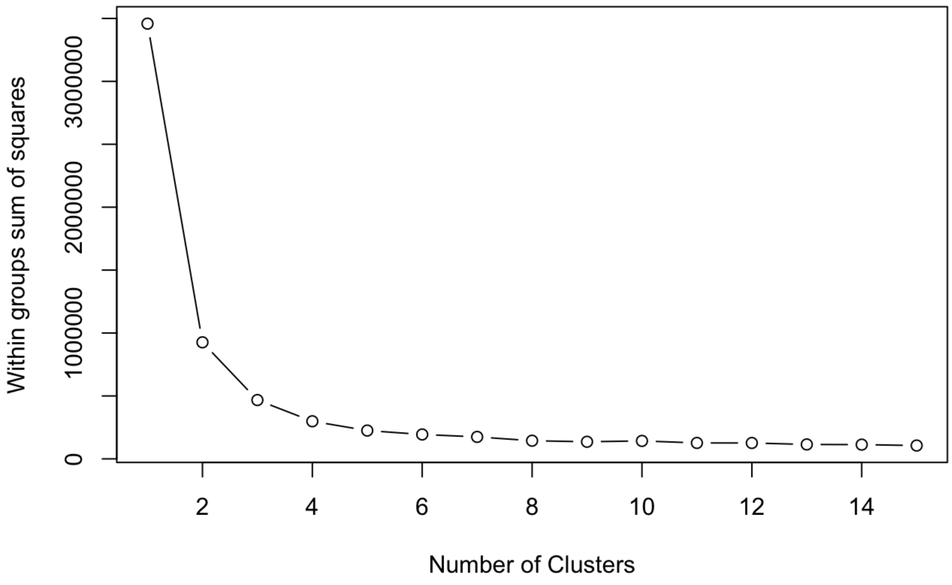
```
q1_model_df$YearStart <- scale(q1_model_df$YearStart);
q1_model_df$Data_Value <- scale(q1_model_df$Data_Value);
q1_model_df$Stratification1 <- scale(q1_model_df$Stratification1);
str(q1_model_df)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 15554 obs. of  7 variables:
##  $ YearStart              : num [1:15554, 1] -1.46 -1.46 -1.46 -1.46 -1.46 ...
##   ..- attr(*, "scaled:center")= num 2013
##   ..- attr(*, "scaled:scale")= num 1.71
##  $ LocationAbbr           : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Class                  : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ Data_Value             : num [1:15554, 1] 0.0134 0.0586 -0.0167 0.2543 0.1338 ...
##   ..- attr(*, "scaled:center")= num 31.9
##   ..- attr(*, "scaled:scale")= num 6.64
##  $ QuestionID             : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ StratificationCategory1: int  6 3 3 2 2 2 2 1 1 1 ...
##  $ Stratification1        : num [1:15554, 1] 1.707 1.096 0.362 0.973 0.606 ...
##   ..- attr(*, "scaled:center")= num 14
##   ..- attr(*, "scaled:scale")= num 8.18
##  - attr(*, "spec")=
##   .. cols(
##   ..   YearStart = col_double(),
##   ..   YearEnd = col_double(),
##   ..   LocationAbbr = col_character(),
##   ..   LocationDesc = col_character(),
##   ..   Datasource = col_character(),
##   ..   Class = col_character(),
##   ..   Topic = col_character(),
##   ..   Question = col_character(),
##   ..   Data_Value_Unit = col_logical(),
##   ..   Data_Value_Type = col_character(),
##   ..   Data_Value = col_double(),
##   ..   Data_Value_Alt = col_double(),
##   ..   Data_Value_Footnote_Symbol = col_character(),
##   ..   Data_Value_Footnote = col_character(),
##   ..   Low_Confidence_Limit = col_double(),
##   ..   High_Confidence_Limit = col_double(),
##   ..   Sample_Size = col_double(),
##   ..   Total = col_character(),
##   ..   `Age(years)` = col_character(),
##   ..   Education = col_character(),
##   ..   Gender = col_character(),
##   ..   Income = col_character(),
##   ..   `Race/Ethnicity` = col_character(),
##   ..   GeoLocation = col_character(),
##   ..   ClassID = col_character(),
##   ..   TopicID = col_character(),
##   ..   QuestionID = col_character(),
##   ..   DataValueTypeID = col_character(),
##   ..   LocationID = col_character(),
##   ..   StratificationCategory1 = col_character(),
##   ..   Stratification1 = col_character(),
##   ..   StratificationCategoryId1 = col_character(),
##   ..   StratificationID1 = col_character()
##   .. )
```

Determine number of clusters

Run tests on various numbers of clusters to look for an “elbow” which will suggest how many useful clusters we can hope to get from this data.

```
wss <- (nrow(q1_model_df)-1)*sum(apply(q1_model_df,2,var))
for (i in 2:15) wss[i] <- sum(kmeans(q1_model_df,
   centers=i)$withinss)
plot(1:15, wss, type="b", xlab="Number of Clusters",
  ylab="Within groups sum of squares")
```

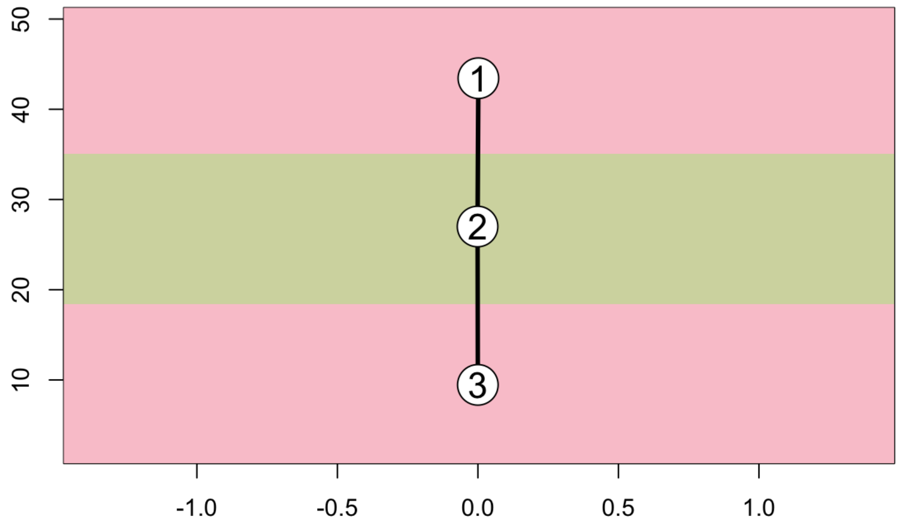


In terms of how many clusters we should choose, these graphs could be suggesting anywhere from 3 to 4, depending on how we read them.

Build a kmean model with 3 clusters:

```
# K-Means Cluster Analysis
fit <- kmeans(q1_model_df, 3) # 3 cluster solution
```

```
# Optional Step - We need to export a predictable model - so that, we will use library flexclust
q1_model_df_shiny_model_1 = kcca(q1_model_df, k=3, kccaFamily("kmeans"));
saveRDS(q1_model_df_shiny_model_1, file = "./cluster-model-1.rda");
image(q1_model_df_shiny_model_1);
```



```
# get cluster means
aggregate(q1_model_df,by=list(fit$cluster),FUN=mean);
```

```
##   Group.1     YearStart LocationAbbr Class  Data_Value QuestionID
## 1       1  3.153973e-03    42.935922     2  0.02399765        3.5
## 2       2  3.303493e-05     8.960199     2 -0.06466337        3.5
## 3       3 -3.170259e-03    26.003090     2  0.04139491        3.5
##   StratificationCategory1 Stratification1
## 1                3.177864   -0.0071063938
## 2                3.204363    0.0060760518
## 3                3.187717    0.0009355893
```

```
# append cluster assignment
q1_model_df <- data.frame(q1_model_df, fit$cluster)
## Assess Model
```

Let’s take a look how many dataset in each clusters. And what is performance of this model.

```
q1_model_df %>% group_by(fit.cluster) %>% summarise(n());
```

```
## # A tibble: 3 x 2
##   fit.cluster `n()`
##         <int> <int>
## 1           1  5150
## 2           2  5226
## 3           3  5178
```

```
fit
```

```
## K-means clustering with 3 clusters of sizes 5150, 5226, 5178
## 
## Cluster means:
##       YearStart LocationAbbr Class  Data_Value QuestionID
## 1  3.153973e-03    42.935922     2  0.02399765        3.5
## 2  3.303493e-05     8.960199     2 -0.06466337        3.5
## 3 -3.170259e-03    26.003090     2  0.04139491        3.5
##   StratificationCategory1 Stratification1
## 1                3.177864   -0.0071063938
## 2                3.204363    0.0060760518
## 3                3.187717    0.0009355893
## 
## Clustering vector:
##     [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##    [35] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##    [69] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [103] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [137] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [171] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [205] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [239] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [273] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [307] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [341] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [375] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [409] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [443] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [477] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [511] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [545] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [579] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [613] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [647] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [681] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [715] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [749] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [783] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [817] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [851] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [885] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [919] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [953] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##   [987] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1021] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1055] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1089] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1123] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1157] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1191] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1225] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1259] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1293] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1327] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1361] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1395] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1429] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1463] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1497] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1531] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1565] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1599] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1633] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1667] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1701] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1735] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1769] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1803] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1837] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1871] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1905] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1939] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [1973] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2007] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2041] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2075] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2109] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2143] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2177] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2211] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2245] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2279] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2313] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2347] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2381] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2415] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2449] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2483] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2517] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2551] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2585] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2619] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2653] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2687] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2721] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2755] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2789] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2823] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2857] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2891] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2925] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2959] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [2993] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3027] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3061] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3095] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3129] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3163] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3197] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3231] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3265] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3299] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3333] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3367] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3401] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3435] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [3469] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3
##  [3503] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3537] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3571] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3605] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3639] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3673] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3707] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3741] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3775] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3809] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3843] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3877] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3911] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3945] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [3979] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4013] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4047] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4081] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4115] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4149] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4183] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4217] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4251] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4285] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4319] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4353] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4387] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4421] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4455] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4489] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4523] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4557] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4591] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4625] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4659] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4693] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4727] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4761] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4795] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4829] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4863] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4897] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4931] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4965] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [4999] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5033] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5067] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5101] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5135] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5169] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5203] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5237] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5271] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5305] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5339] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5373] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5407] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5441] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5475] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5509] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5543] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5577] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5611] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5645] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5679] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5713] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5747] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5781] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5815] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5849] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5883] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5917] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5951] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [5985] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6019] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6053] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6087] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6121] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6155] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6189] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6223] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6257] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6291] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6325] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6359] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6393] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6427] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6461] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6495] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6529] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6563] 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [6597] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [6631] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [6665] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [6699] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [6733] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [6767] 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6801] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6835] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6869] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6903] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6937] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [6971] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [7005] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [7039] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 3 3 3 3 3
##  [7073] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [7107] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
##  [7141] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1
##  [7175] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7209] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7243] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7277] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7311] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7345] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7379] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7413] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7447] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7481] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7515] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7549] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7583] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7617] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7651] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7685] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7719] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7753] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7787] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7821] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7855] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7889] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7923] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7957] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [7991] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8025] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8059] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8093] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8127] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8161] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8195] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8229] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8263] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8297] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8331] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8365] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8399] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8433] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8467] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8501] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8535] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8569] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8603] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8637] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8671] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8705] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8739] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8773] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8807] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8841] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8875] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8909] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8943] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [8977] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9011] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9045] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9079] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9113] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9147] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9181] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9215] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9249] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9283] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9317] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9351] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9385] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9419] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9453] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9487] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9521] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9555] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9589] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9623] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9657] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9691] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9725] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9759] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9793] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9827] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9861] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9895] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9929] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9963] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [9997] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10031] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10065] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10099] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10133] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10167] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10201] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10235] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10269] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10303] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10337] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [10371] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
## [10405] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10439] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10473] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10507] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10541] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10575] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10609] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10643] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10677] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10711] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10745] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10779] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10813] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10847] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10881] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10915] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10949] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [10983] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [11017] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [11051] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [11085] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [11119] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [11153] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [11187] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [11221] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [11255] 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11289] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11323] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11357] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11391] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11425] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11459] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11493] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11527] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11561] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11595] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11629] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11663] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11697] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11731] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11765] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11799] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11833] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11867] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11901] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11935] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [11969] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [12003] 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12037] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3
## [12071] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [12105] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [12139] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1
## [12173] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12207] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12241] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12275] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12309] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12343] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12377] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12411] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12445] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12479] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12513] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12547] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12581] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12615] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12649] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12683] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12717] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12751] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12785] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12819] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12853] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12887] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12921] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [12955] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [12989] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3
## [13023] 3 3 3 3 3 3 3 1 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2
## [13057] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [13091] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2
## [13125] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [13159] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [13193] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [13227] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [13261] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2
## [13295] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2
## [13329] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [13363] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [13397] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [13431] 2 2 3 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13465] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13499] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13533] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13567] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13601] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13635] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13669] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13703] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13737] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13771] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13805] 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3
## [13839] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [13873] 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [13907] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [13941] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [13975] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [14009] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [14043] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [14077] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [14111] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [14145] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [14179] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [14213] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [14247] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 3 2 2 2 2 2 2 2 2
## [14281] 2 2 2 2 2 2 2 2 3 3 3 3 3 3 1 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2
## [14315] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14349] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14383] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14417] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14451] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14485] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14519] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14553] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14587] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14621] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14655] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14689] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [14723] 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [14757] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [14791] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [14825] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [14859] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [14893] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [14927] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [14961] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [14995] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [15029] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [15063] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1
## [15097] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3
## [15131] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
## [15165] 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15199] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15233] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15267] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15301] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15335] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15369] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15403] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15437] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15471] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15505] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [15539] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## 
## Within cluster sum of squares by cluster:
## [1] 152611.1 155858.7 155860.3
##  (between_SS / total_SS =  86.6 %)
## 
## Available components:
## 
## [1] "cluster"      "centers"      "totss"        "withinss"    
## [5] "tot.withinss" "betweenss"    "size"         "iter"        
## [9] "ifault"
```

This model has a high cluster mean 86% (between_SS / total_SS). 86% is a measure of the total variance in your data set that is explained by the clustering.

Build Hierarchical Model #2

Let’s build a simple clustering model. We need to convert factor to numeric

```
set.seed(786);
q1_h_model_df <- q1;
q1_h_model_df$Sample_Size <- NULL;
q1_h_model_df$LocationAbbr <- as.integer(q1_h_model_df$LocationAbbr);
q1_h_model_df$Class <- as.integer(q1_h_model_df$Class);
q1_h_model_df$QuestionID <- as.integer(q1_h_model_df$QuestionID);
q1_h_model_df$Stratification1 <- as.integer(q1_h_model_df$Stratification1);
q1_h_model_df$StratificationCategory1 <- as.integer(q1_h_model_df$StratificationCategory1)
```

Scaling data

Use R’s scale() function to scale all your column values instead of one by one.

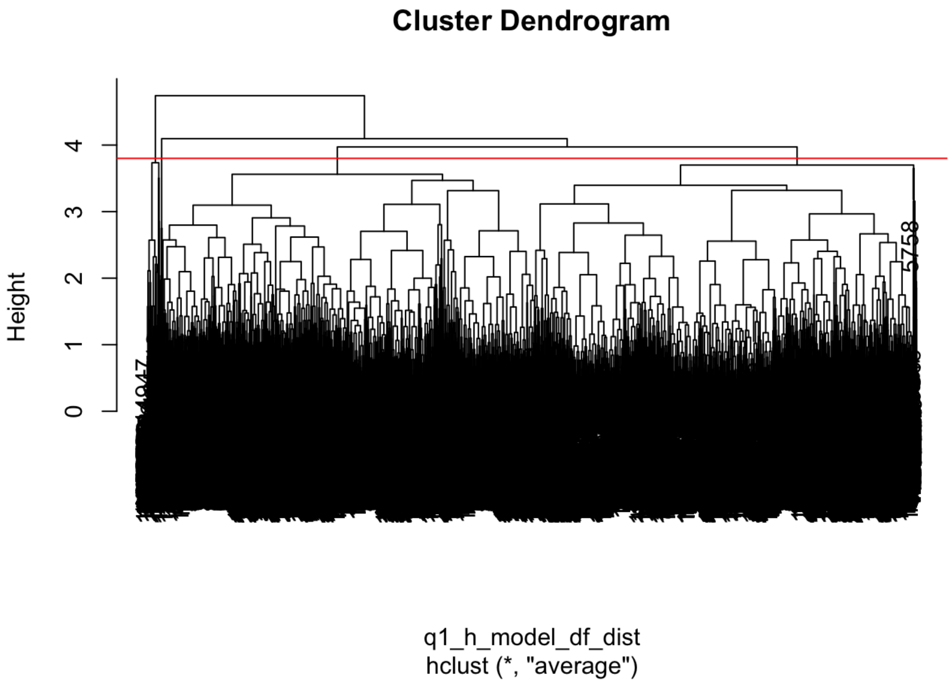
```
q1_h_model_df <- as.data.frame(scale(q1_h_model_df))
```

Calculate euclidean distance

```
q1_h_model_df_dist <- dist(q1_h_model_df, method = 'euclidean')
```

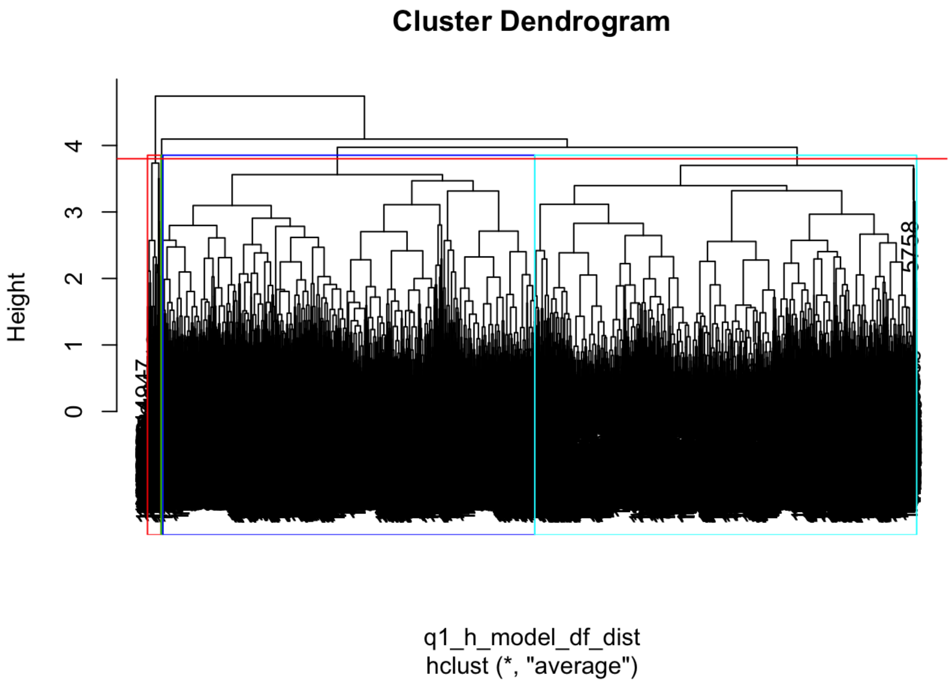
Build a hierarchical model

```
hclust_avg <- hclust(q1_h_model_df_dist, method = 'average');
plot(hclust_avg);
abline(h = 3.8, col = 'red')
```



As you can see in the dengrogram if we cut the tree at 4 (h=4), we get 3 clusters. However, we can see there is 2 big clusters just right under around h=3.8, if cut the tree at 3.8, we would get 4 clusters. And it also match the “elbow” chart. In the 1st model, we choose k=3. In this dengrogram, let’s cut the tree at 3.8, so that, we will get 4 clusters.

```
cut_avg <- cutree(hclust_avg, h = 3.8);
plot(hclust_avg);
rect.hclust(hclust_avg , k = 4, border = 2:6);
abline(h = 3.8, col = 'red')
```



However, we can see the 2nd cluster(between green and blue) one, which is too small. It might make sence we should only use k=3 rather than 4.

Supervised Learning

Build Linear Regression Model #1

Lets start using the clean data set and removing the Class question as it has no effect on the model since we are applying the training only for Obesity / Weight Status

```
df_su <- q1

df_su$Class <- NULL
df_su$QuestionID <- NULL
df_su$StratificationCategory1 <- NULL
#data.frame(df_su)

split = sample.split(df_su$Data_Value, SplitRatio = 0.7)
trainingset = subset(df_su, split == TRUE)
testset = subset(df_su, split == FALSE)
```

Fitting Simple Linear Regression to the Training set

```
lm.r= lm(formula = Data_Value ~.,
                        data = trainingset)
coef(lm.r)
```

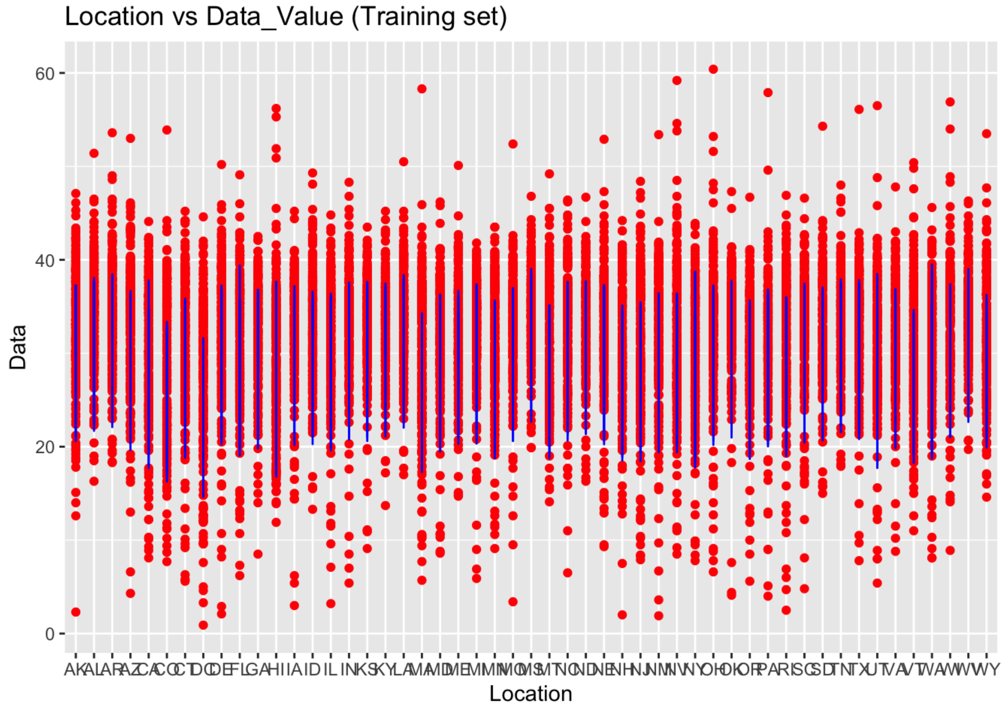
```
##                                      (Intercept) 
##                                    -3.934636e+02 
##                                        YearStart 
##                                     2.121545e-01 
##                                   LocationAbbrAL 
##                                     6.717537e-01 
##                                   LocationAbbrAR 
##                                     1.079240e+00 
##                                   LocationAbbrAZ 
##                                    -7.422335e-01 
##                                   LocationAbbrCA 
##                                    -2.714090e+00 
##                                   LocationAbbrCO 
##                                    -4.114303e+00 
##                                   LocationAbbrCT 
##                                    -1.564250e+00 
##                                   LocationAbbrDC 
##                                    -5.732333e+00 
##                                   LocationAbbrDE 
##                                    -1.116415e-01 
##                                   LocationAbbrFL 
##                                    -1.318251e+00 
##                                   LocationAbbrGA 
##                                    -5.743092e-01 
##                                   LocationAbbrHI 
##                                    -3.696375e+00 
##                                   LocationAbbrIA 
##                                    -2.091531e-01 
##                                   LocationAbbrID 
##                                    -7.511159e-01 
##                                   LocationAbbrIL 
##                                    -9.811723e-01 
##                                   LocationAbbrIN 
##                                     1.602654e-01 
##                                   LocationAbbrKS 
##                                     2.154857e-01 
##                                   LocationAbbrKY 
##                                     6.402129e-02 
##                                   LocationAbbrLA 
##                                     1.002355e+00 
##                                   LocationAbbrMA 
##                                    -3.102120e+00 
##                                   LocationAbbrMD 
##                                    -9.361702e-01 
##                                   LocationAbbrME 
##                                    -7.310928e-01 
##                                   LocationAbbrMI 
##                                    -5.447497e-02 
##                                   LocationAbbrMN 
##                                    -1.605747e+00 
##                                   LocationAbbrMO 
##                                    -4.057320e-01 
##                                   LocationAbbrMS 
##                                     1.645522e+00 
##                                   LocationAbbrMT 
##                                    -2.232428e+00 
##                                   LocationAbbrNC 
##                                     2.870356e-01 
##                                   LocationAbbrND 
##                                     3.273741e-01 
##                                   LocationAbbrNE 
##                                    -1.295837e-01 
##                                   LocationAbbrNH 
##                                    -2.066749e+00 
##                                   LocationAbbrNJ 
##                                    -1.943423e+00 
##                                   LocationAbbrNM 
##                                    -9.660375e-01 
##                                   LocationAbbrNV 
##                                    -9.752618e-01 
##                                   LocationAbbrNY 
##                                    -2.616703e+00
##                                   LocationAbbrOH
##                                    -1.865191e-01
##                                   LocationAbbrOK
##                                     5.975650e-01
##                                   LocationAbbrOR
##                                    -1.702652e+00
##                                   LocationAbbrPA
##                                    -5.630933e-01
##                                   LocationAbbrRI
##                                    -1.393586e+00
##                                   LocationAbbrSC
##                                     3.880276e-02
##                                   LocationAbbrSD
##                                    -3.349759e-01
##                                   LocationAbbrTN
##                                     5.348165e-01
##                                   LocationAbbrTX
##                                     4.186301e-01
##                                   LocationAbbrUT
##                                    -2.666684e+00
##                                   LocationAbbrVA
##                                    -5.439188e-01
##                                   LocationAbbrVT
##                                    -2.759458e+00
##                                   LocationAbbrWA
##                                    -1.635333e+00
##                                   LocationAbbrWI
##                                     1.288003e-02
##                                   LocationAbbrWV
##                                     1.628861e+00
##                                   LocationAbbrWY
##                                    -1.114007e+00
##                                      Sample_Size
##                                     4.000593e-05
##                 Stratification1$25,000 - $34,999
##                                     2.463100e-01
##                 Stratification1$35,000 - $49,999
##                                     6.847568e-01
##                 Stratification1$50,000 - $74,999
##                                     5.413461e-01
##                Stratification1$75,000 or greater
##                                    -8.438214e-01
##                           Stratification118 - 24
##                                    -1.216546e+01
##                   Stratification12 or more races
##                                    -9.106586e-01
##                           Stratification125 - 34
##                                    -2.509534e+00
##                           Stratification135 - 44
##                                     1.187546e+00
##                           Stratification145 - 54
##                                     2.574572e+00
##                           Stratification155 - 64
##                                     3.086428e+00
##                       Stratification165 or older
##                                     4.400863e-01
##     Stratification1American Indian/Alaska Native
##                                     2.015200e+00
##                             Stratification1Asian
##                                    -1.280316e+01
##                  Stratification1College graduate
##                                    -3.197859e+00
##                 Stratification1Data not reported
##                                    -4.605389e+00
##                            Stratification1Female
##                                    -4.155407e+00
##         Stratification1Hawaiian/Pacific Islander
##                                     7.106816e+00
##              Stratification1High school graduate
##                                     1.705950e-01
##                          Stratification1Hispanic
##                                          6.998277e-01
##                Stratification1Less than $15,000
##                                         -1.050818e+00
##            Stratification1Less than high school
##                                          5.923528e-01
##                             Stratification1Male
##                                          2.187950e+00
##               Stratification1Non-Hispanic Black
##                                          2.718624e+00
##               Stratification1Non-Hispanic White
##                                         -1.465170e+00
##                            Stratification1Other
##                                         -1.232487e+00
## Stratification1Some college or technical school
##                                         -3.023579e-01
##                            Stratification1Total
##                                         -1.244653e+00
```

Predicting the Test set results

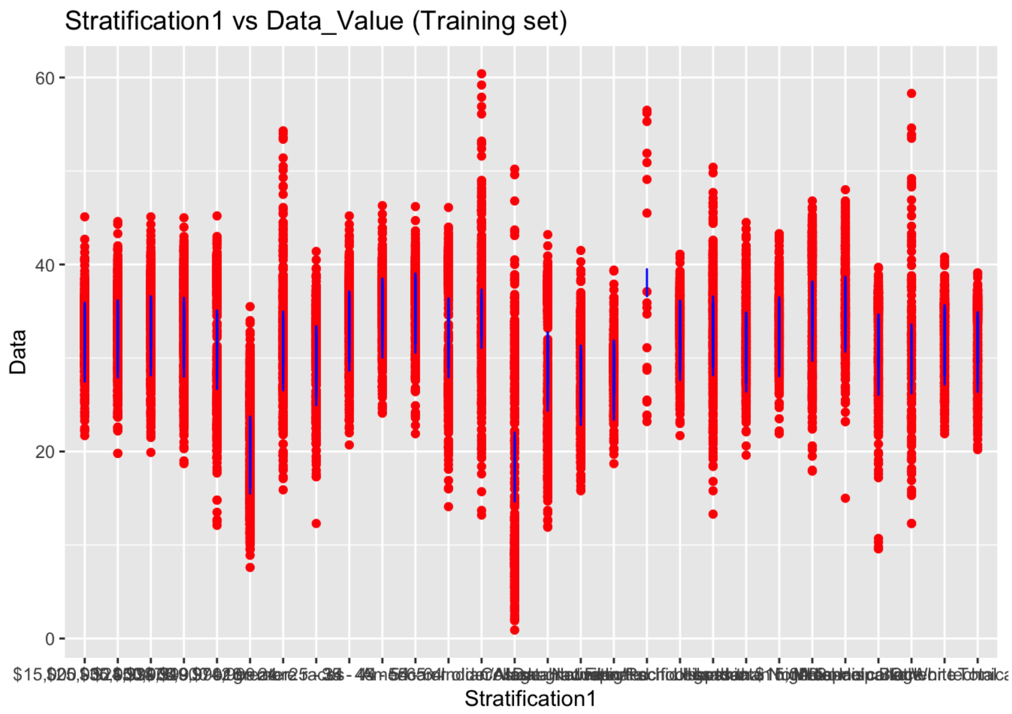
```
ypred = predict(lm.r, newdata = testset)  -> df_result
```

Visualising the Training set results

```
ggplot() + geom_point(aes(x = trainingset$LocationAbbr,
                y = trainingset$Data_Value), colour = 'red') +
geom_line(aes(x = trainingset$LocationAbbr,
y = predict(lm.r, newdata = trainingset)), colour = 'blue') +

ggtitle('Location vs Data_Value (Training set)') +
xlab('Location') +
ylab('Data')
```

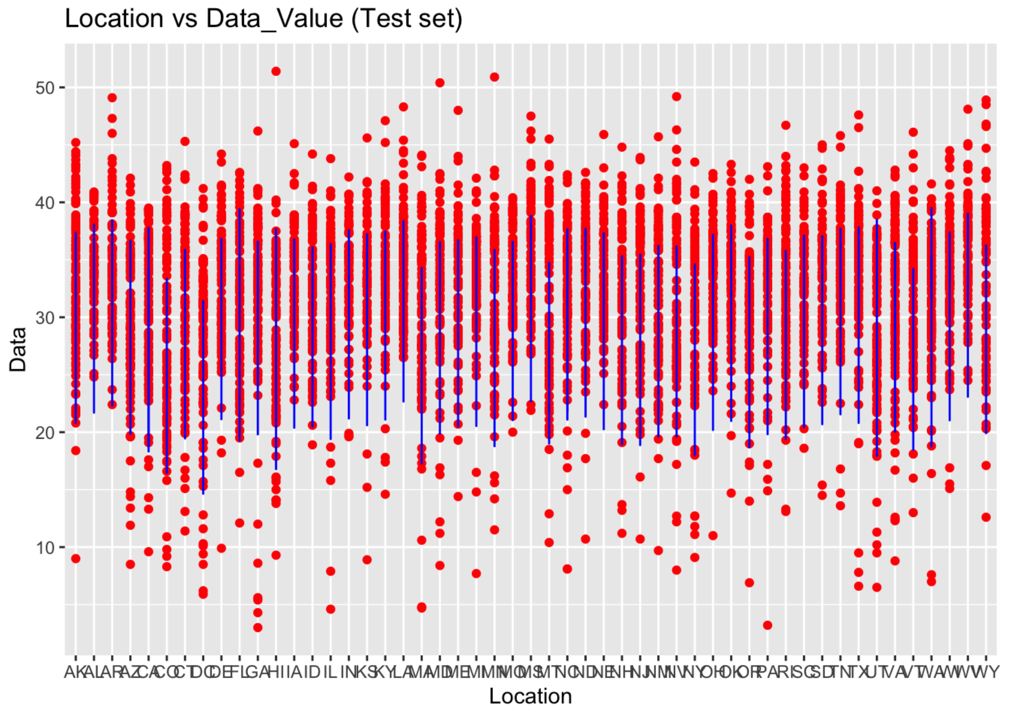


```
ggplot() + geom_point(aes(x = trainingset$Stratification1 ,
                y = trainingset$Data_Value), colour = 'red') +
geom_line(aes(x = trainingset$Stratification1,
y = predict(lm.r, newdata = trainingset)), colour = 'blue') +

ggtitle('Stratification1 vs Data_Value (Training set)') +
xlab('Stratification1') +
ylab('Data')
```



Visualising the Test set results

```
  ggplot() + geom_point(aes(x = testset$LocationAbbr,
              y = testset$Data_Value), colour = 'red') +
geom_line(aes(x = testset$LocationAbbr,
y = predict(lm.r, newdata = testset)), colour = 'blue') +

ggtitle('Location vs Data_Value (Test set)') +
xlab('Location') +
ylab('Data')
```



Finding the error and root mean square of error

```
cbind(Actual=testset$Data_Value,Predicted=df_result)->Final_Data

as.data.frame(Final_Data)->Final_Data


(Final_Data$Actual- Final_Data$Predicted)->error

cbind(Final_Data,error)->Final_Data

rmse1<-sqrt(mean(Final_Data$error^2))

rmse1
```

```
## [1] 5.413371
```

Model extraction

```
saveRDS(lm.r, file = "./q2_model_df_su.rda");
```

Build Random Forest Model #2

Prepare data

only select YearStart, LocationAbbr, Data_Value, StratificationCategory1 and Stratification1 columns.

```
df_su_model_2 <- q1[,c(1,2,4,7,8)];
```

Split the data into Training dataset and Test dataset

Take 75% of the dataset for training data, 25% for test data

```
smp_size <- floor(0.75 * nrow(df_su_model_2));
set.seed(123);
train_ind <- sample(seq_len(nrow(df_su_model_2)), size = smp_size);
df_su_model_2_train <- df_su_model_2[train_ind, ];
df_su_model_2_test <- df_su_model_2[-train_ind, ];
```

Define the equation & create a model

```
df_su_model_2_rf <- randomForest(Data_Value ~., data=df_su_model_2_train, ntree=2000, importance=TRUE)
```

Check randmon forset model performance

```
df_su_model_2_rf
```

```
## 
## Call:
##  randomForest(formula = Data_Value ~ ., data = df_su_model_2_train,      ntree = 2000, importance = TRUE) 
##                Type of random forest: regression
##                      Number of trees: 2000
## No. of variables tried at each split: 1
## 
##           Mean of squared residuals: 33.56577
##                     % Var explained: 23.33
```

We can see “Mean of squared residuals: 33.56577”, which tells us our model accuracy is accuracy 66.44%.

Evalutaion

```
result <- data.frame(df_su_model_2_test$Data_Value, predict(df_su_model_2_rf, df_su_model_2_test[,], type="response"))
```

Let’s take a look the predict value vs the actual value

```
head(result,5)
```

```
##   df_su_model_2_test.Data_Value
## 1                          31.8
## 2                          16.3
## 3                          40.1
## 4                          32.9
## 5                          30.5
##   predict.df_su_model_2_rf..df_su_model_2_test......type....response..
## 1                                                             30.94641
## 2                                                             24.41382
## 3                                                             33.85643
## 4                                                             32.97992
## 5                                                             30.94641
```

It is not bad. Some records' prediciton is very close. For example, row 1, actural value is 31.8. The prediction value is 30.824.

Export Model

```
saveRDS(df_su_model_2_rf, file = "./randomforest_model.rda")
```

Accordint to the graph we have above, we knew this obesity rate has a different behavior compare to each catetory. If we only choose one category to build our model, will it has a better output?

Build Random Forest Model #3

Prepare data

Only select YearStart, LocationAbbr, Data_Value, StratificationCategory1 and Stratification1 columns. And filter out catetory equal to "Income"

```
df_su_model_3 <- q1[,c(1,2,4,7,8)];
df_su_model_3  <- df_su_model_3 %>% filter(StratificationCategory1 == "Income")
```

Split the data into Training dataset and Test dataset

Take 75% of the dataset for training data, 25% for test data

```
smp_size <- floor(0.75 * nrow(df_su_model_3));
set.seed(123);
train_ind <- sample(seq_len(nrow(df_su_model_3)), size = smp_size);
df_su_model_3_train <- df_su_model_3[train_ind, ];
df_su_model_3_test <- df_su_model_3[-train_ind, ];
```

Define the equation & create a model

```
df_su_model_3_rf <- randomForest(Data_Value ~., data=df_su_model_3_train, ntree=2000, importance=TRUE)
```

Check randmon forset model performance

```
df_su_model_3_rf
```

```
## 
## Call:
##  randomForest(formula = Data_Value ~ ., data = df_su_model_3_train,      ntree = 2000, importance = TRUE) 
##                Type of random forest: regression
##                      Number of trees: 2000
## No. of variables tried at each split: 1
## 
##           Mean of squared residuals: 25.36678
##                     % Var explained: 9.32
```

We can see "Mean of squared residuals: 25.36678", which tells us our model accuracy is accuracy 74.63%. It is better than the previous model.

Evalutaion

```
result <- data.frame(df_su_model_3_test$Data_Value, predict(df_su_model_3_rf, df_su_model_3_test[,], type="response"))
```

Let's take a look the predict value vs the actual value

```
head(result,5)
```

```
##   df_su_model_3_test.Data_Value
## 1                          35.8
## 2                          28.8
## 3                          36.0
## 4                          28.2
## 5                          31.8
##   predict.df_su_model_3_rf..df_su_model_3_test......type....response..
## 1                                                             33.03106
## 2                                                             32.76699
## 3                                                             30.76949
## 4                                                             33.02681
## 5                                                             33.09700
```

Let’s take a look the 1st row again. Actural value is 35.8. The prediction value is 33.03106. It is much closer than previous model. And this one also improve if we only build the model based on different category, we can get a better model.

Conclusions

We wrangled this large, complicated, messy dataset into a much more compact entity representing only data we are interested in. Through our exploration of the data, we were able to see the widely-known correlations between a number of factors and obesity rate. These factors include age, gender, education, ethnic background, income, quality of diet, and quantity of exercise.

Through unsupervised learning, we discovered that some demographics appear to set themselves apart, both in terms of demographic and behavioural features, and observed obesity rates. We discovered which features were most strongly correlated with obesity overall. This information can be useful both for predicting which groups stand most to benefit from behavioural changes, as well as giving us clues about which behavioural changes are most relevant to those groups.

The knowledge that was discovered in this process, in combination with the problem we set out to solve, contributed to the design and development of the user interface for our Shiny app, which can be found here: https://ptgray.shinyapps.io/project/ (https://ptgray.shinyapps.io/project/)

Recommendations

A predictive model, based on supervised learning using obesity rate as the value to predict, is essential to the completion of this project and is the only recommendation we have at this time.

Ideally, we would train, evaluate, and optimize a model that can predict obesity rate for a given demographic (age, gender, level of education) using known values for Exercise and Eating Scores. We would validate this predicted obesity rate by comparing it to the known value of the obesity rate for that group. We would then predict a selection of hypothetical obesity rates for this demographic using artifical values representing modest (~10%) improvements in a selection of behaviours. This would give us insights into which behaviours, or combination of behaviours, would be of most value in an obesity reduction campaign. Finally, we would compare these theoretical obesity rates with the observed obesity rate for the comparable demographic, but with a higher level of education. From our exploration, there seems to be a good chance that the message “stay in school!” will be of more utility than any tinkering with people’s lifestyles after the fact.

CSML1000 York University “Machine Learning in the Business Context”. Fall 2019

Final Project, Group 8

Tamer Hanna tamerh@my.yorku.ca (mailto:tamerh@my.yorku.ca) Pete Gray ptgray@my.yorku.ca (mailto:ptgray@my.yorku.ca) Xiaohai Lu yu271637@my.yorku.ca (mailto:yu271637@my.yorku.ca) Haofeng Zhou zhf85@my.yorku.ca (mailto:zhf85@my.yorku.ca)

.