SRM Exercise

Bartu Kadir Tamer

2025-06-27

Uploading the Related Packages:

```
required_packages <- c(
  "ggplot2", "Hmisc", "car", "caret", "MASS", "randomForest",
  "xgboost", "tidyverse", "tidyr", "dplyr", "rpart", "rpart.plot",
  "ipred", "e1071"
)

# Install missing packages only
install_if_missing <- function(pkg) {
  if (!requireNamespace(pkg, quietly = TRUE)) {
    install.packages(pkg, dependencies = TRUE)
  }
}

# Loop through and install if needed
invisible(lapply(required_packages, install_if_missing))
```

```
library(ggplot2)
library(Hmisc) #Used for descriptive statistics
```

```
##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
library(car)
```

```
## Loading required package: carData
```

```
library(caret)
```

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

```r
library(MASS)
library(randomForest)
```

```
## randomForest 4.7-1.2

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

##
## Attaching package: 'randomForest'

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

```r
library(xgboost)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::combine()       masks randomForest::combine()
## x dplyr::filter()        masks stats::filter()
## x dplyr::lag()           masks stats::lag()
## x purrr::lift()          masks caret::lift()
## x randomForest::margin() masks ggplot2::margin()
## x dplyr::recode()        masks car::recode()
## x dplyr::select()        masks MASS::select()
## x dplyr::slice()         masks xgboost::slice()
## x purrr::some()          masks car::some()
## x dplyr::src()           masks Hmisc::src()
## x dplyr::summarize()     masks Hmisc::summarize()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(tidyr)
library(dplyr)
library(rpart)
library(rpart.plot)
library(ipred)
library(e1071)
```

```
##
## Attaching package: 'e1071'
##
## The following object is masked from 'package:Hmisc':
##
##     impute
```

To understand working directory of the Rstudio. It is important to have the file which we want to analyze in same directory.

```r
getwd()
```

```
## [1] "/Users/ben/Downloads/SRM Exercise/SRM Exercise"
```

If the Downloads file contatins the dataset you can copy the path and paste it into the read.csv() function.

Uploading and Viewing the Dataset:

```r
df=read.csv("EIB2023_dummies_filtered.csv",header = T)
View(df)
```

Data Summary:

Variables :

progr_binary: Support for progressive taxation on carbon consumption (rich pay more, poor pay less on fuel taxes)

ctax_binary (Response variable for this exercise): Support for taxing environmentally harmful profits (e.g., from fossil fuels)

subsidies_binary: Support for removing fossil fuel subsidies and redirecting funds to renewables

income_scale: Country-normalized income score (standardized by mean and SD)

any_cc_last2year_factor: Whether the respondent's region had major climate disasters in the last 2 years (yes/no)

gender: Respondent's gender

educationsecondary: Dummy = 1 if respondent completed secondary education

educationtertiary: Dummy = 1 if respondent completed tertiary education

age: Respondent's age (can be used directly or scaled as age_scale)

urbanizationtown: Dummy = 1 if respondent lives in a town/suburban area

has_children: Dummy = 1 if respondent has children

trust: Respondent's trust in their country's ability to fight climate change and maintain social equity Levels: Very confident, Rather confident, Not really confident, Not confident at all

lr_scale: Left–right political ideology scale (0 = far left, 10 = far right)

new_ccknowledge_index: Composite score of climate change knowledge

countryBelgium, countryFrance, . . . , countrySweden, etc. 0/1 dummies for each country

str() function helps to understand dimension and structures pf the objects in the dataset.

```r
str(df)
```

```
## 'data.frame':	22729 obs. of  47 variables:
##  $ progr_binary                : int  1 0 1 1 1 1 1 1 1 0 ...
##  $ ctax_binary                 : int  1 1 1 1 1 1 1 1 1 0 ...
##  $ subsidies_binary            : int  1 1 1 1 1 1 1 1 1 0 ...
##  $ income_scale                : num  -1.403 NA -0.446 0.484 1.057 ...
##  $ age_scale                   : num  -9.54 -31.54 -17.54 5.46 20.46 ...
##  $ trust                       : chr  "No really confident" "Rather confident" "No really confident"
##  $ LR_scale_scale              : num  4.348 1.348 0.348 1.348 -1.652 ...
##  $ new_ccknowledge_index_scale: num  0.0249 -0.3084 0.0249 0.136 0.0249 ...
##  $ any_cc_last2year_factor    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ regional_heterogeneity     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ LR_scale                   : int  10 7 6 7 4 3 4 NA 2 NA ...
##  $ income                     : chr  "Less than 7 070 €" "Prefer not to say" "14 230 € to under 16 54
##  $ new_ccknowledge_index      : num  0.667 0.333 0.667 0.778 0.667 ...
##  $ gender                     : chr  "male" "male" "female" "female" ...
##  $ country_w                  : num  0.962 1.134 0.934 0.911 1.009 ...
##  $ educationprimary           : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ educationsecondary         : int  1 1 1 0 1 0 0 0 0 1 ...
##  $ educationtertiary          : int  0 0 0 1 0 1 1 1 1 0 ...
##  $ urbanizationtown           : int  0 1 1 0 0 1 1 0 1 1 ...
##  $ urbanizationrural          : int  1 0 0 0 1 0 0 0 0 0 ...
##  $ has_childrenyes            : int  0 0 0 0 0 1 1 0 0 0 ...
##  $ countryBelgium             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryBulgaria            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryCroatia             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryCyprus              : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryCzech.Republic      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryDenmark             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryEstonia             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryFinland             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryFrance              : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryGermany             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryGreece              : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryHungary             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryIreland             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryItaly               : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ countryLatvia              : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryLithuania           : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryLuxembourg          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryMalta               : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryPoland              : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryPortugal            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryRomania             : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countrySlovakia            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countrySlovenia            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countrySpain               : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countrySweden              : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ countryThe.Netherlands     : int  0 0 0 0 0 0 0 0 0 0 ...
```

Counting the number of NA values.

```
sum(is.na(df))
```

```
## [1] 7064
```

We have 7064 NA values in the dataset. Some of the ML function that we are using the packages causes error when we want to deal with NA generated dataset. Therefore, it is better to getting rid of them.

```
df_clean=na.omit(df)
sum(is.na(df_clean))
```

```
## [1] 0
```

We will use the ctax_binary as our response in this analysis, so we can delete the other binary responses.

```
df_clean=df_clean[,-c(1,3,12)]
```

We can check the descriptive statistics of the dataset for understanding the first insights of the dataset.

```
describe(df_clean)
```

```
## df_clean
##
##  44  Variables      18698  Observations
## --------------------------------------------------------------------------------
## ctax_binary
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.524    14479   0.7744
##
## --------------------------------------------------------------------------------
## income_scale
##          n    missing   distinct       Info       Mean    pMedian        Gmd
##      18698          0        270          1    0.03855    0.03021      1.143
##        .05        .10        .25        .50        .75        .90        .95
## -1.3884036 -1.3140793 -0.9228093  0.0004226  1.0398237  1.3125329  1.4025175
##
## lowest : -1.77838 -1.64296 -1.61668 -1.51722 -1.49367
## highest: 1.57411  1.61239  1.66954  1.67938  1.79275
## --------------------------------------------------------------------------------
## age_scale
##        n  missing distinct     Info     Mean  pMedian      Gmd      .05
##    18698        0     1767        1    1.145    1.105     19.2 -25.4740
##      .10      .25      .50      .75      .90      .95
## -22.4740 -12.4740   0.9247  15.5260  23.1522  26.4630
##
## lowest : -31.5352 -31.474  -30.593  -30.5669 -30.561
## highest: 45.4678  46.1922  47.463   51.4331  53.664
## --------------------------------------------------------------------------------
## trust
##        n  missing distinct
##    18698        0        4
##
## Value      No confident at all No really confident    Rather confident
## Frequency                 3016                8373                5763
## Proportion               0.161               0.448               0.308
##
## Value           Very confident
```

```
## Frequency                  1546
## Proportion                0.083
## ---------------------------------------------------------------------------
## LR_scale_scale
##        n  missing distinct     Info     Mean  pMedian      Gmd      .05
##    18698        0      270        1  0.01449 -0.05924    2.395  -3.6809
##      .10      .25      .50      .75      .90      .95
##  -2.6809  -1.1092  -0.3348   1.3482   2.9913   4.2339
##
## lowest : -5       -4.878   -4.84746 -4.83556 -4.82637
## highest: 4.70789  4.89079  4.94488  4.98488  4.99129
## ---------------------------------------------------------------------------
## new_ccknowledge_index_scale
##        n  missing distinct     Info     Mean  pMedian      Gmd      .05
##    18698        0      412        1  0.01096  0.01894   0.1795 -0.27900
##      .10      .25      .50      .75      .90      .95
## -0.21584 -0.09509  0.02644  0.12713  0.20194  0.24044
##
## lowest : -0.533278 -0.510613 -0.508222 -0.499557 -0.497397
## highest: 0.380189  0.381778  0.381889  0.39527   0.40071
## ---------------------------------------------------------------------------
## any_cc_last2year_factor
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.738     8174   0.4372
##
## ---------------------------------------------------------------------------
## regional_heterogeneity
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.587    13701   0.7328
##
## ---------------------------------------------------------------------------
## LR_scale
##        n  missing distinct     Info     Mean  pMedian      Gmd      .05
##    18698        0       10    0.966    5.546      5.5    2.387        2
##      .10      .25      .50      .75      .90      .95
##        3        4        5        7        8       10
##
## Value          1     2     3     4     5     6     7     8     9    10
## Frequency    750   702  1584  1847  5612  2599  1985  1767   799  1053
## Proportion 0.040 0.038 0.085 0.099 0.300 0.139 0.106 0.095 0.043 0.056
##
## For the frequency table, variable is rounded to the nearest 0
## ---------------------------------------------------------------------------
## new_ccknowledge_index
##        n  missing distinct     Info     Mean  pMedian      Gmd      .05
##    18698        0       17    0.989   0.6643   0.6667   0.1826   0.3889
##      .10      .25      .50      .75      .90      .95
##   0.4444   0.5556   0.6667   0.7778   0.8333   0.8889
##
## 0.111111111111111 (1, 0.000), 0.166666666666667 (18, 0.001), 0.222222222222222
## (90, 0.005), 0.277777777777778 (275, 0.015), 0.333333333333333 (485, 0.026),
## 0.388888888888889 (713, 0.038), 0.444444444444444 (954, 0.051), 0.5 (1354,
## 0.072), 0.555555555555556 (1576, 0.084), 0.611111111111111 (2021, 0.108),
## 0.666666666666667 (2400, 0.128), 0.722222222222222 (2565, 0.137),
```

```
## 0.777777777777778 (2399, 0.128), 0.833333333333333 (2085, 0.112),
## 0.888888888888889 (1057, 0.057), 0.944444444444444 (544, 0.029), 1 (161, 0.009)
##
## For the frequency table, variable is rounded to the nearest 0
## ---------------------------------------------------------------------------
## gender
##        n  missing distinct
##    18698        0        2
##
## Value      female   male
## Frequency    9244   9454
## Proportion  0.494  0.506
## ---------------------------------------------------------------------------
## country_w
##        n  missing distinct     Info     Mean  pMedian      Gmd      .05
##    18698        0     6967        1   0.9945   0.9657   0.2611   0.6620
##      .10      .25      .50      .75      .90      .95
##   0.7496   0.8595   0.9545   1.0721   1.2468   1.4222
##
## lowest : 0.158708 0.160538 0.174153 0.237714 0.243678
## highest: 3.73711  3.75487  3.77212  4.85878  6.25698
## ---------------------------------------------------------------------------
## educationprimary
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.359     2596   0.1388
##
## ---------------------------------------------------------------------------
## educationsecondary
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.724     7595   0.4062
##
## ---------------------------------------------------------------------------
## educationtertiary
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.744     8507    0.455
##
## ---------------------------------------------------------------------------
## urbanizationtown
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.711     7217    0.386
##
## ---------------------------------------------------------------------------
## urbanizationrural
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.476     3696   0.1977
##
## ---------------------------------------------------------------------------
## has_childrenyes
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.672     6327   0.3384
##
## ---------------------------------------------------------------------------
## countryBelgium
##        n  missing distinct     Info      Sum     Mean
```

```
##    18698        0        2    0.119      775  0.04145
##
## ---------------------------------------------------------------------------
## countryBulgaria
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2     0.13      848  0.04535
##
## ---------------------------------------------------------------------------
## countryCroatia
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.124      809  0.04327
##
## ---------------------------------------------------------------------------
## countryCyprus
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.055      352  0.01883
##
## ---------------------------------------------------------------------------
## countryCzech.Republic
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.129      842  0.04503
##
## ---------------------------------------------------------------------------
## countryDenmark
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.126      820  0.04385
##
## ---------------------------------------------------------------------------
## countryEstonia
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.065      414  0.02214
##
## ---------------------------------------------------------------------------
## countryFinland
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.129      842  0.04503
##
## ---------------------------------------------------------------------------
## countryFrance
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2     0.12      781  0.04177
##
## ---------------------------------------------------------------------------
## countryGermany
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.136      887  0.04744
##
## ---------------------------------------------------------------------------
## countryGreece
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.133      868  0.04642
##
## ---------------------------------------------------------------------------
## countryHungary
```

```
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.118      770  0.04118
##
## ---------------------------------------------------------------------------
## countryIreland
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.132      860  0.04599
##
## ---------------------------------------------------------------------------
## countryItaly
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.121      789   0.0422
##
## ---------------------------------------------------------------------------
## countryLatvia
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.064      406  0.02171
##
## ---------------------------------------------------------------------------
## countryLithuania
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.065      412  0.02203
##
## ---------------------------------------------------------------------------
## countryLuxembourg
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.056      354  0.01893
##
## ---------------------------------------------------------------------------
## countryMalta
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.027      167 0.008931
##
## ---------------------------------------------------------------------------
## countryPoland
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.128      834   0.0446
##
## ---------------------------------------------------------------------------
## countryPortugal
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.131      854  0.04567
##
## ---------------------------------------------------------------------------
## countryRomania
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.133      866  0.04632
##
## ---------------------------------------------------------------------------
## countrySlovakia
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.065      412  0.02203
##
## ---------------------------------------------------------------------------
```

```
## countrySlovenia
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.066      420  0.02246
##
## --------------------------------------------------------------------------------
## countrySpain
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.134      876  0.04685
##
## --------------------------------------------------------------------------------
## countrySweden
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2    0.129      844  0.04514
##
## --------------------------------------------------------------------------------
## countryThe.Netherlands
##        n  missing distinct     Info      Sum     Mean
##    18698        0        2     0.12      779  0.04166
##
## --------------------------------------------------------------------------------
```

EDA Part:

Plotting the Dataset:

What are the distributions of the variables in the dataset ?

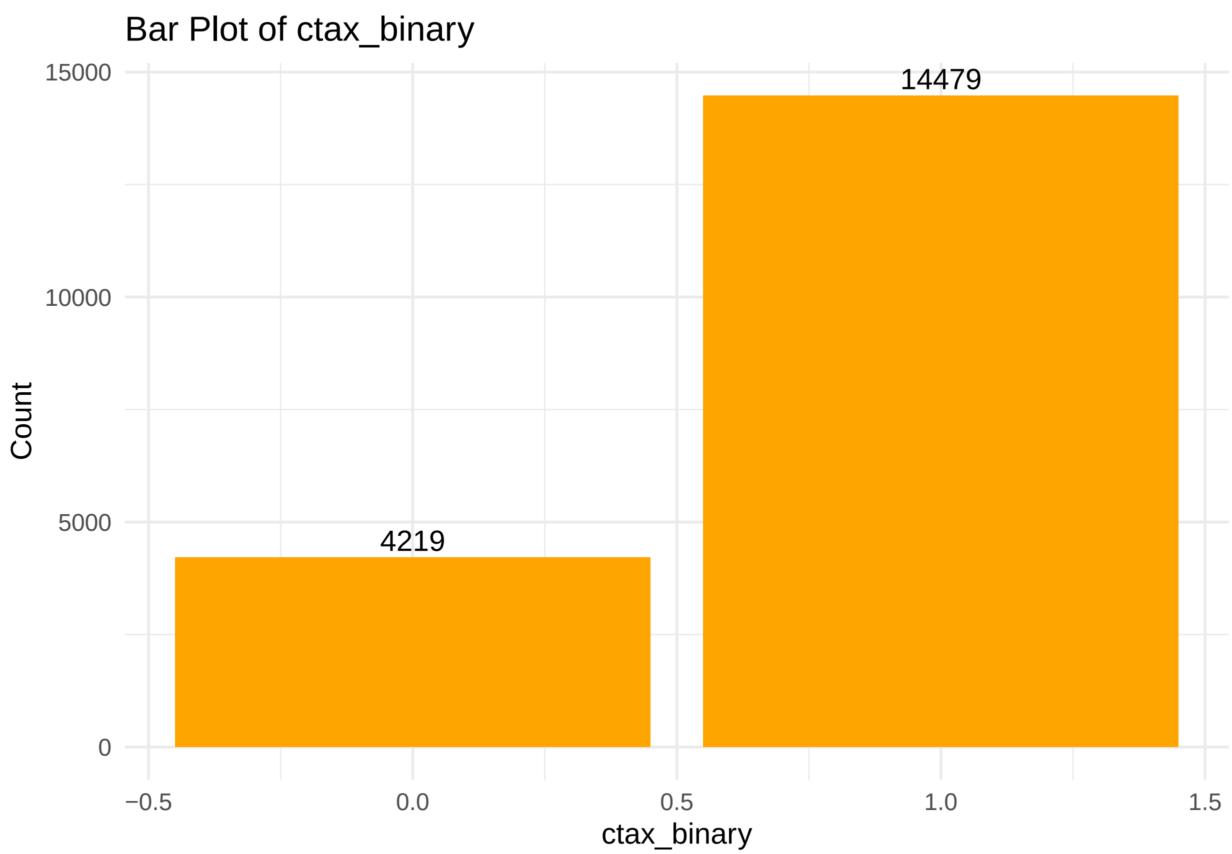
```r
library(ggplot2)

distribution_plot <- function(data) {
  for (v in names(data)) {

    # Skip constant or all-NA variables
    if (all(is.na(data[[v]])) || length(unique(data[[v]])) <= 1) next

    # Histogram for continuous numeric variables
    if (is.numeric(data[[v]]) && length(unique(data[[v]])) > 2) {
      p <- ggplot(data, aes_string(x = v)) +
        geom_histogram(bins = 30, fill = "purple", color = "white") +
        labs(title = paste("Histogram of", v), x = v, y = "Count") +
        theme_minimal()

    # Bar plot for categorical or binary variables
    } else {
      p <- ggplot(data, aes_string(x = v)) +
        geom_bar(fill = "orange") +
        geom_text(stat = "count", aes(label = ..count..), vjust = -0.3) +
        labs(title = paste("Bar Plot of", v), x = v, y = "Count") +
        theme_minimal()
    }

    print(p)
```

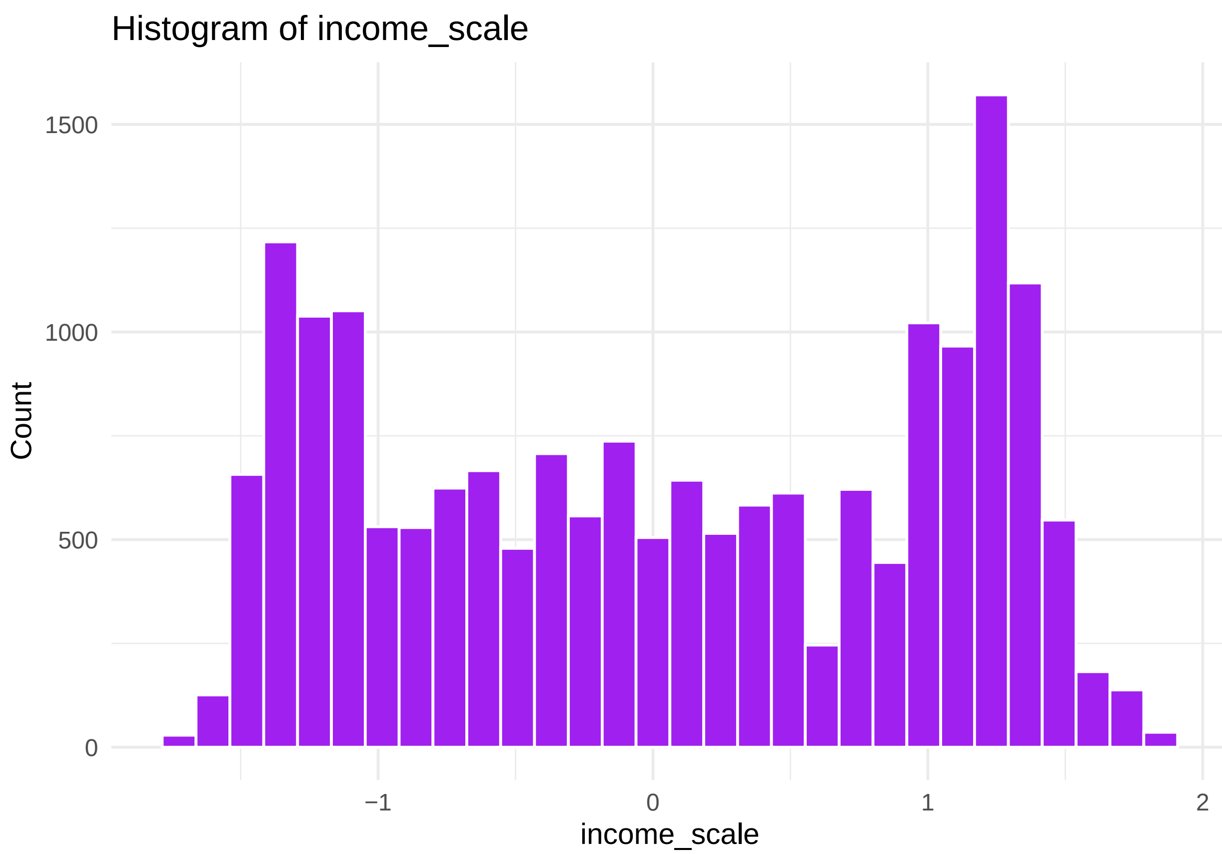
```
  }
}
```

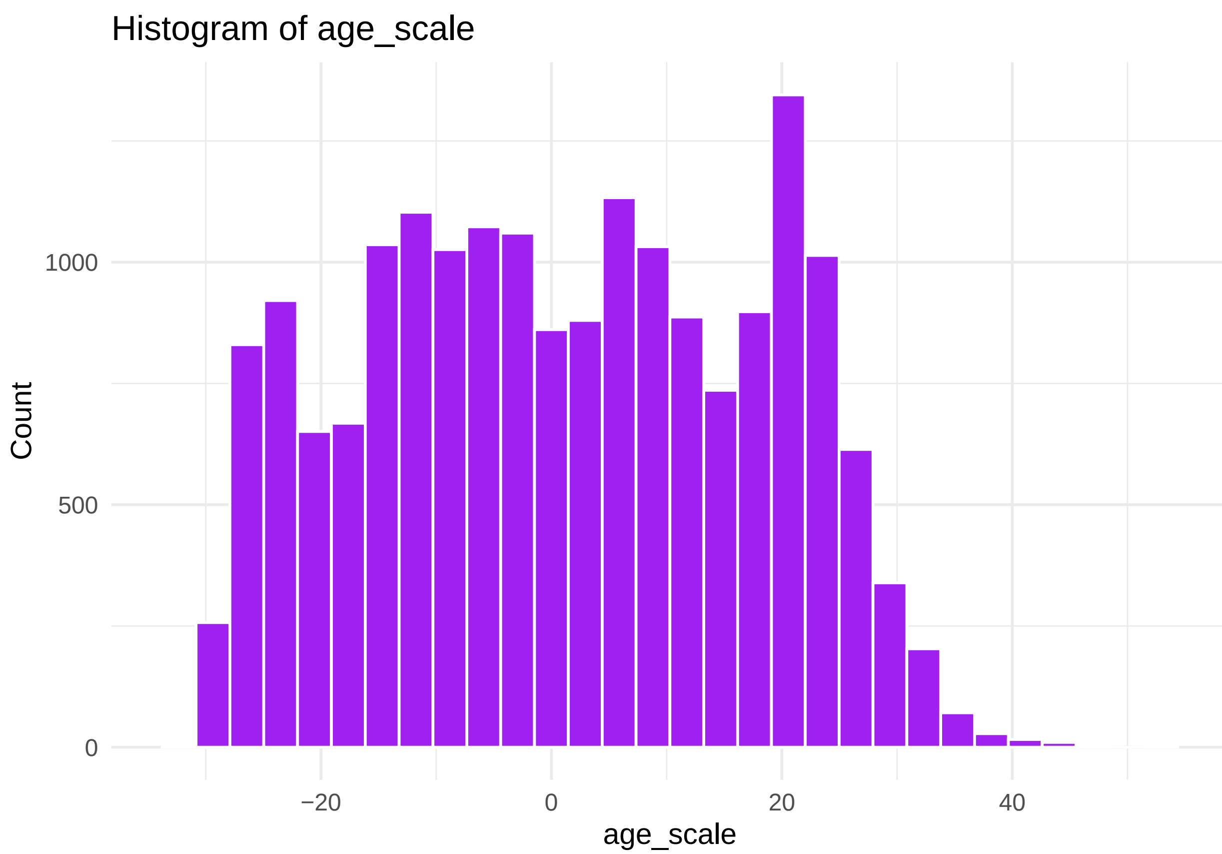
```r
distribution_plot(df_clean)
```

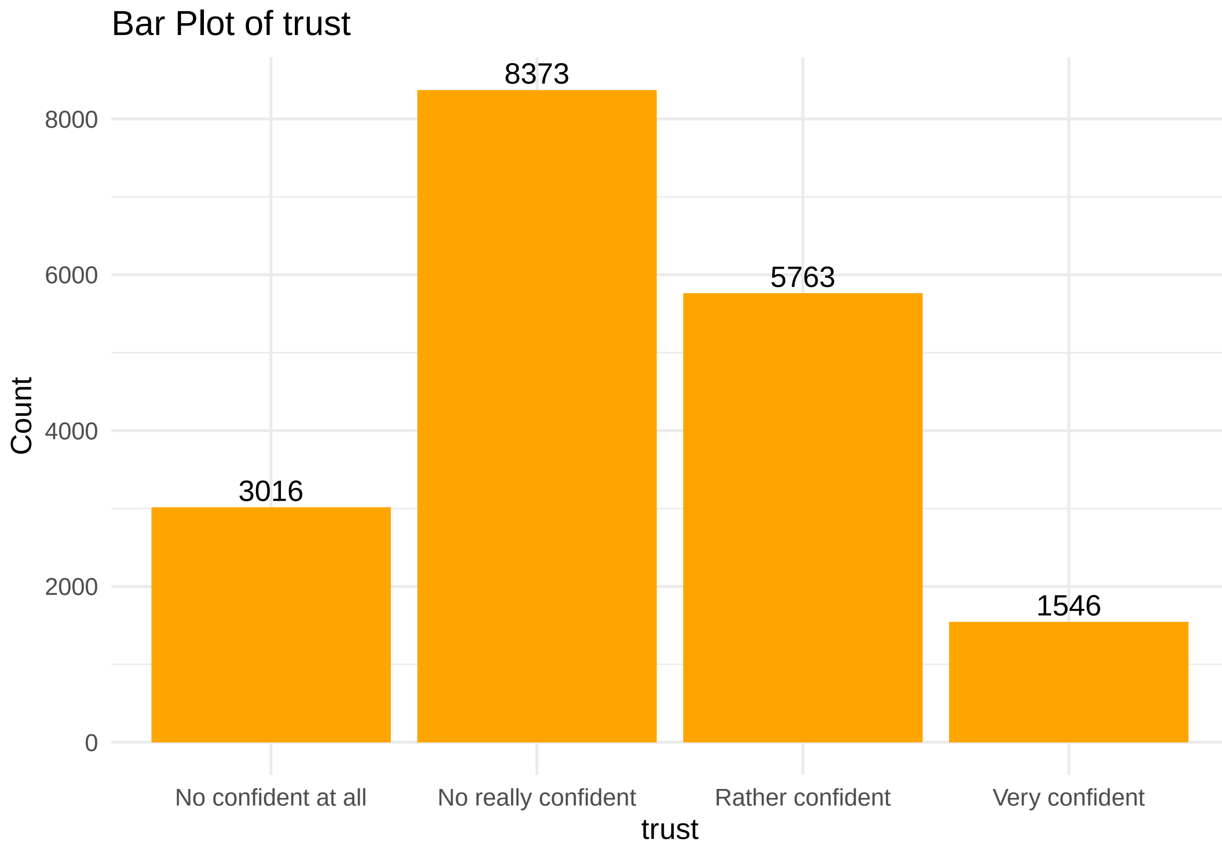
```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

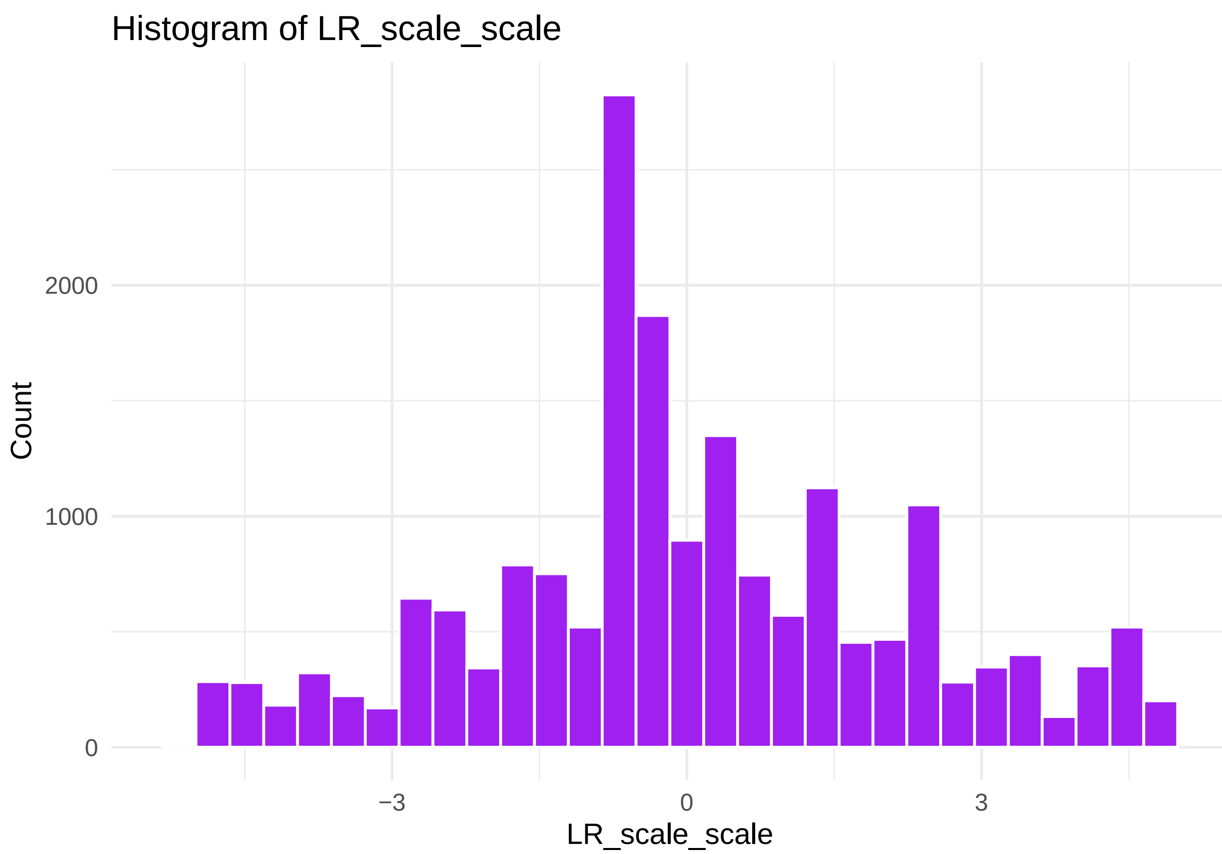
```
## Warning: The dot-dot notation (`..count..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(count)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

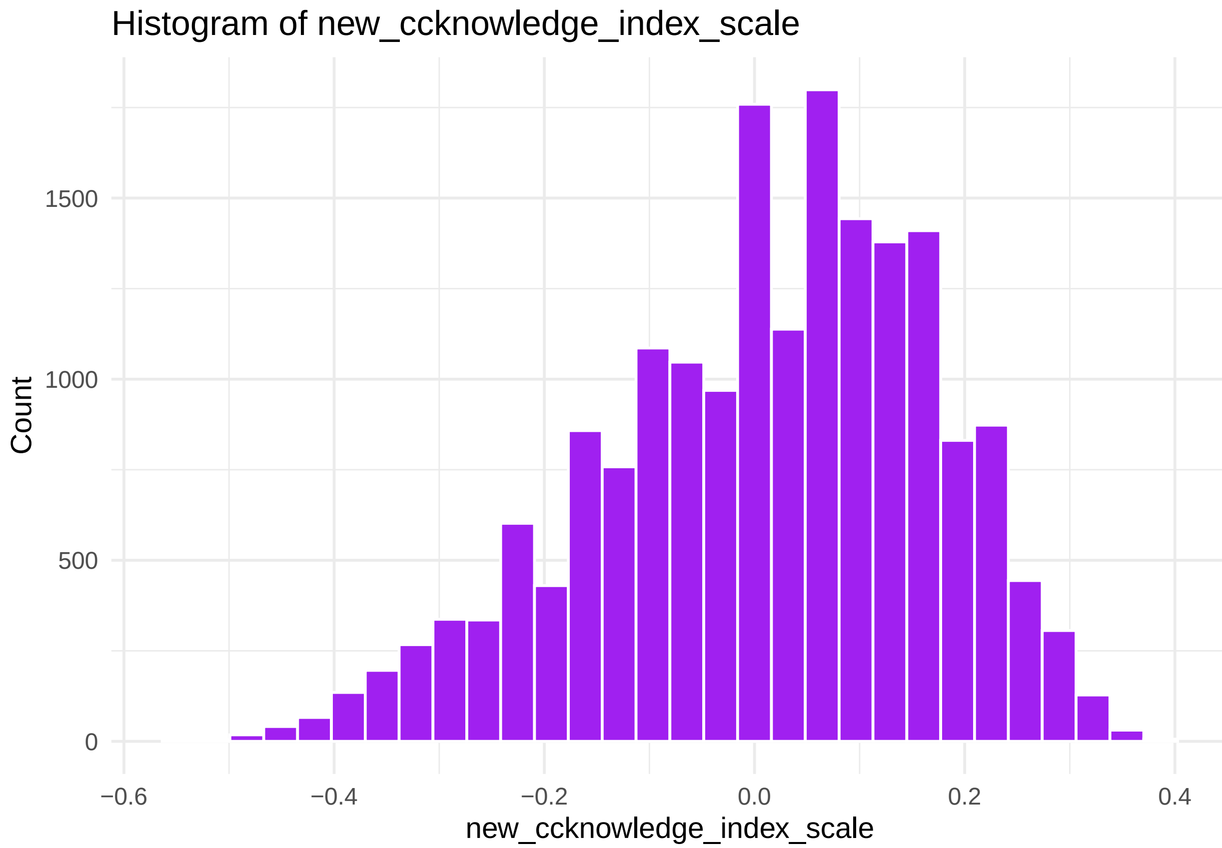



Histogram of income_scale
Count
1500
1000
500
0
−1
0
1
2
income_scale

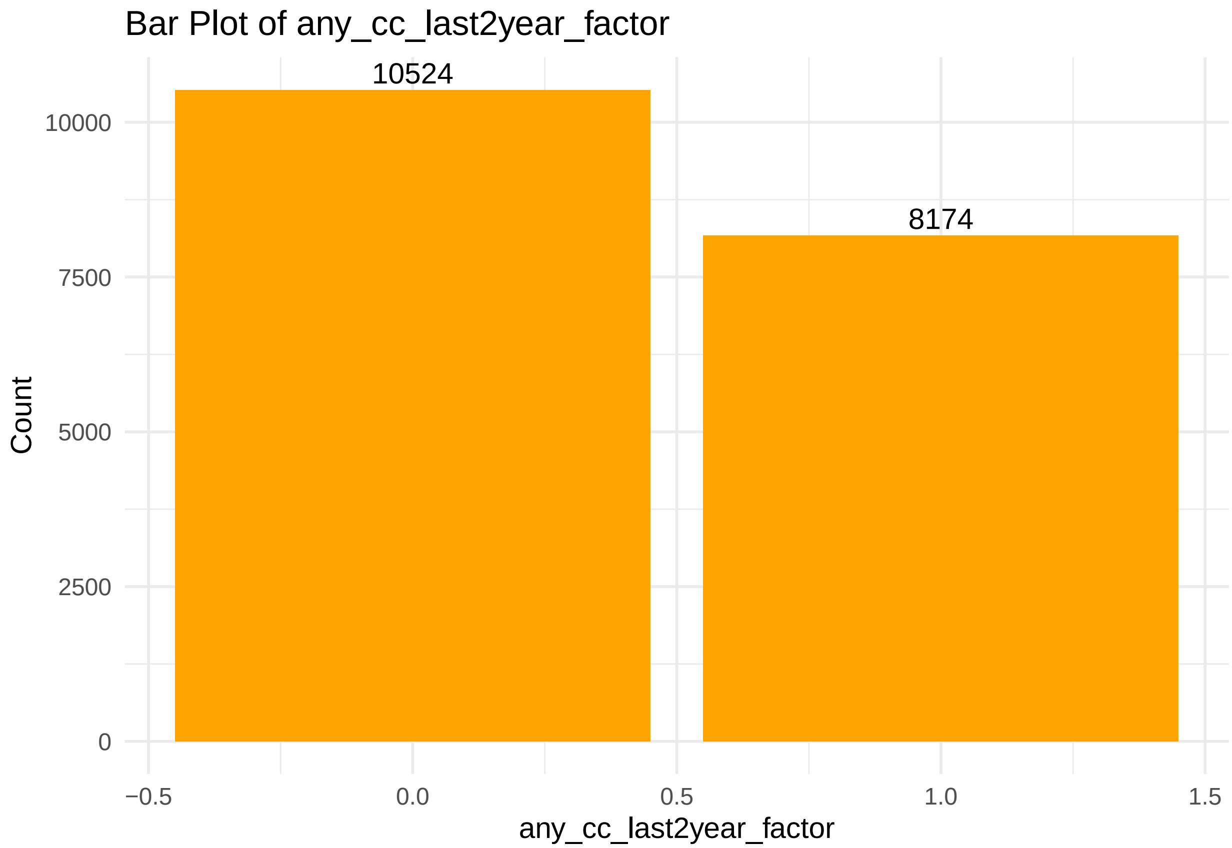

Histogram of age_scale
Count
1000
500
0
−20
0
20
40
age_scale


Bar Plot of trust
8373
5763
3016
1546
8000
6000
4000
2000
0
Count
No confident at all
No really confident
Rather confident
Very confident
trust

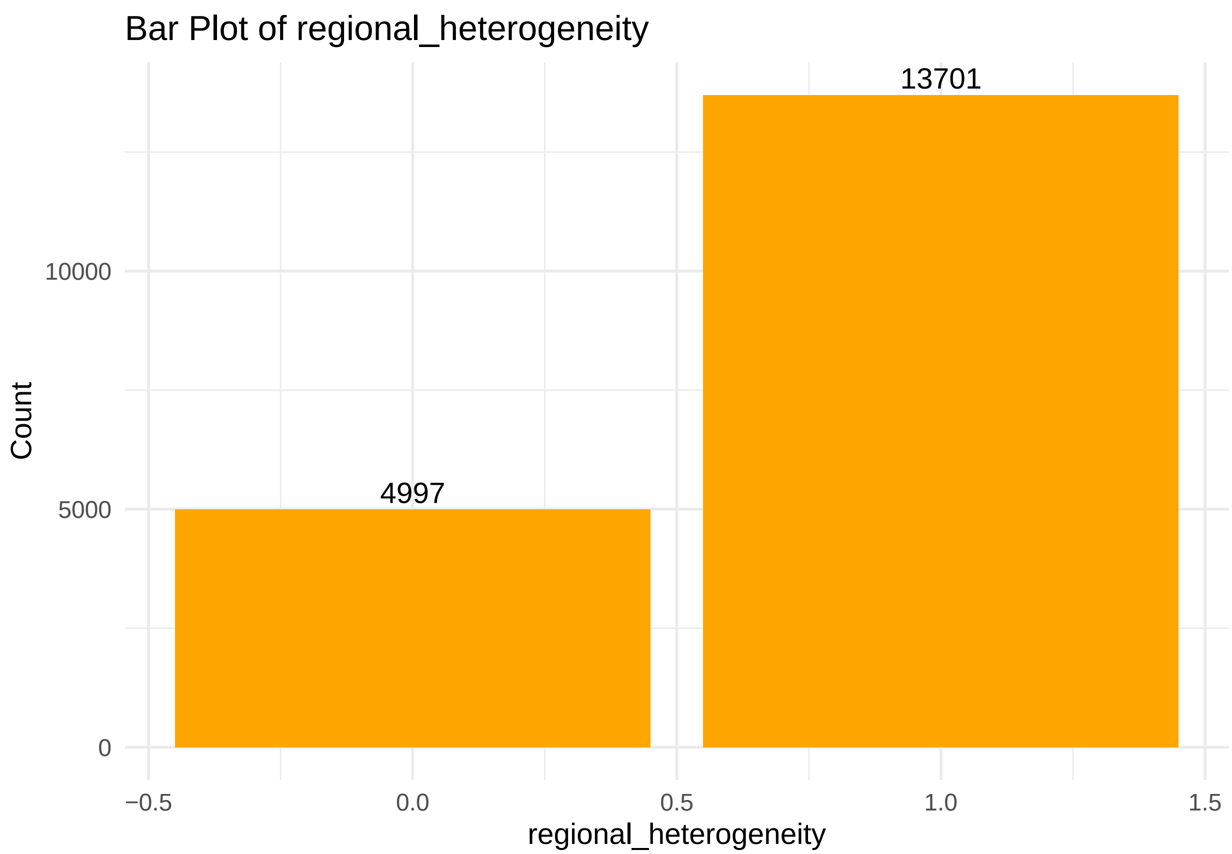

Histogram of LR_scale_scale
Count
2000
1000
0
−3
0
3
LR_scale_scale

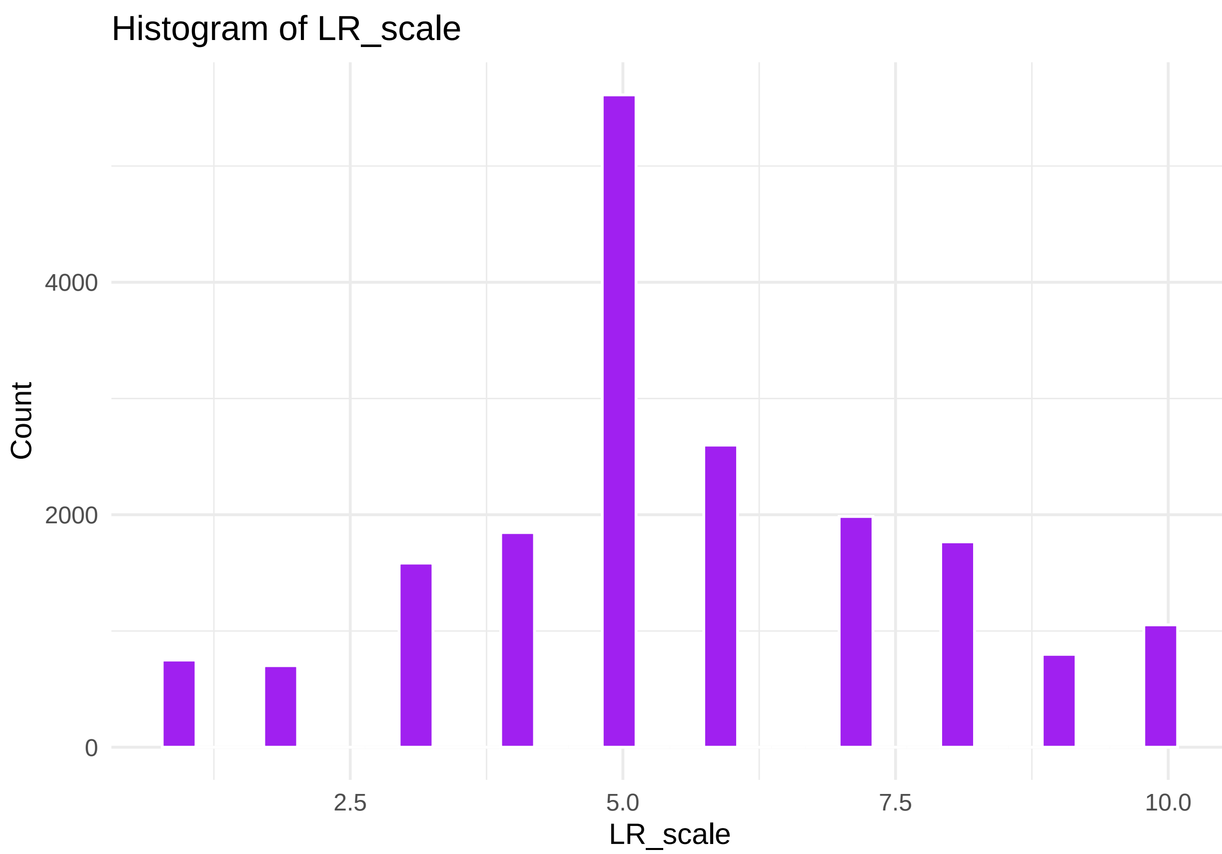


Histogram of new_ccknowledge_index_scale
Count
1500
1000
500
0
−0.6
−0.4
−0.2
0.0
0.2
0.4
new_ccknowledge_index_scale

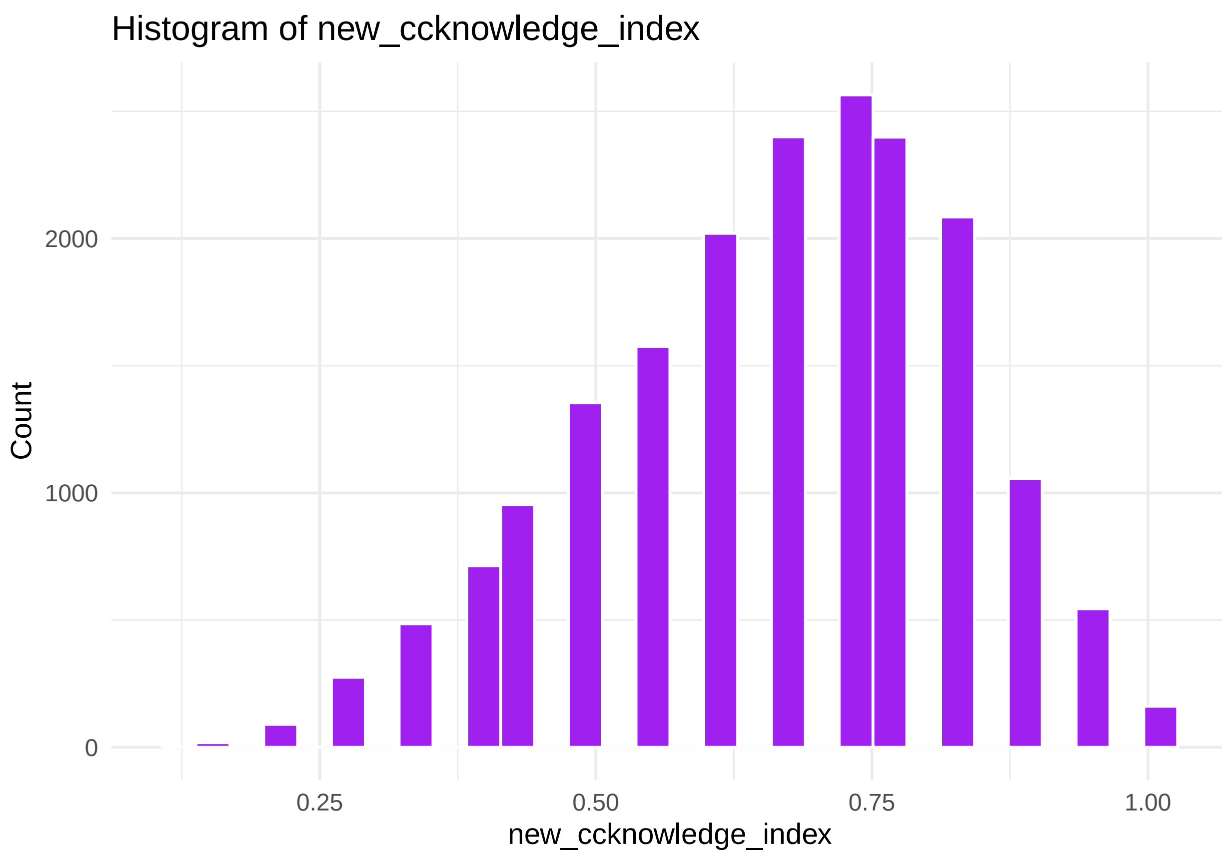


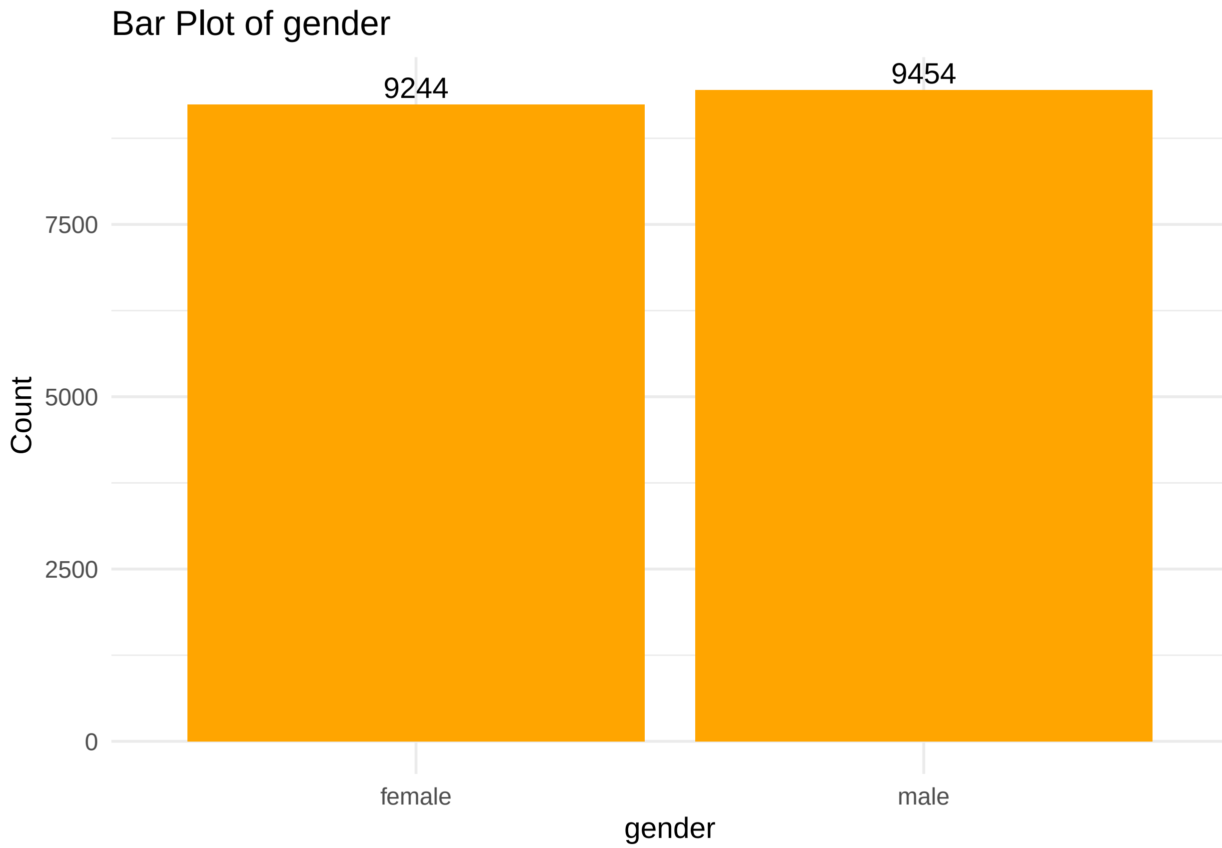
Bar Plot of any_cc_last2year_factor
10524
8174
10000
7500
5000
2500
0
Count
−0.5
0.0
0.5
1.0
1.5
any_cc_last2year_factor

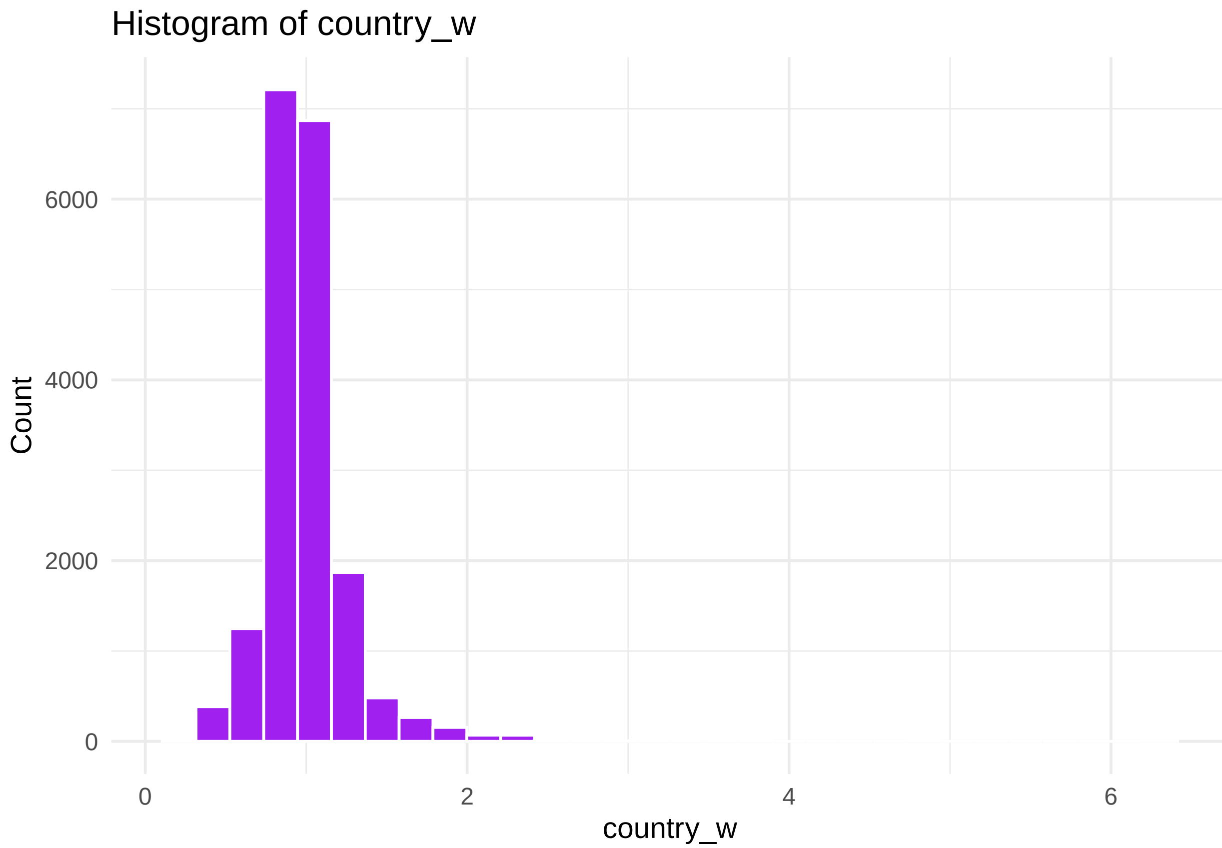

Bar Plot of regional_heterogeneity
13701
4997
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
regional_heterogeneity

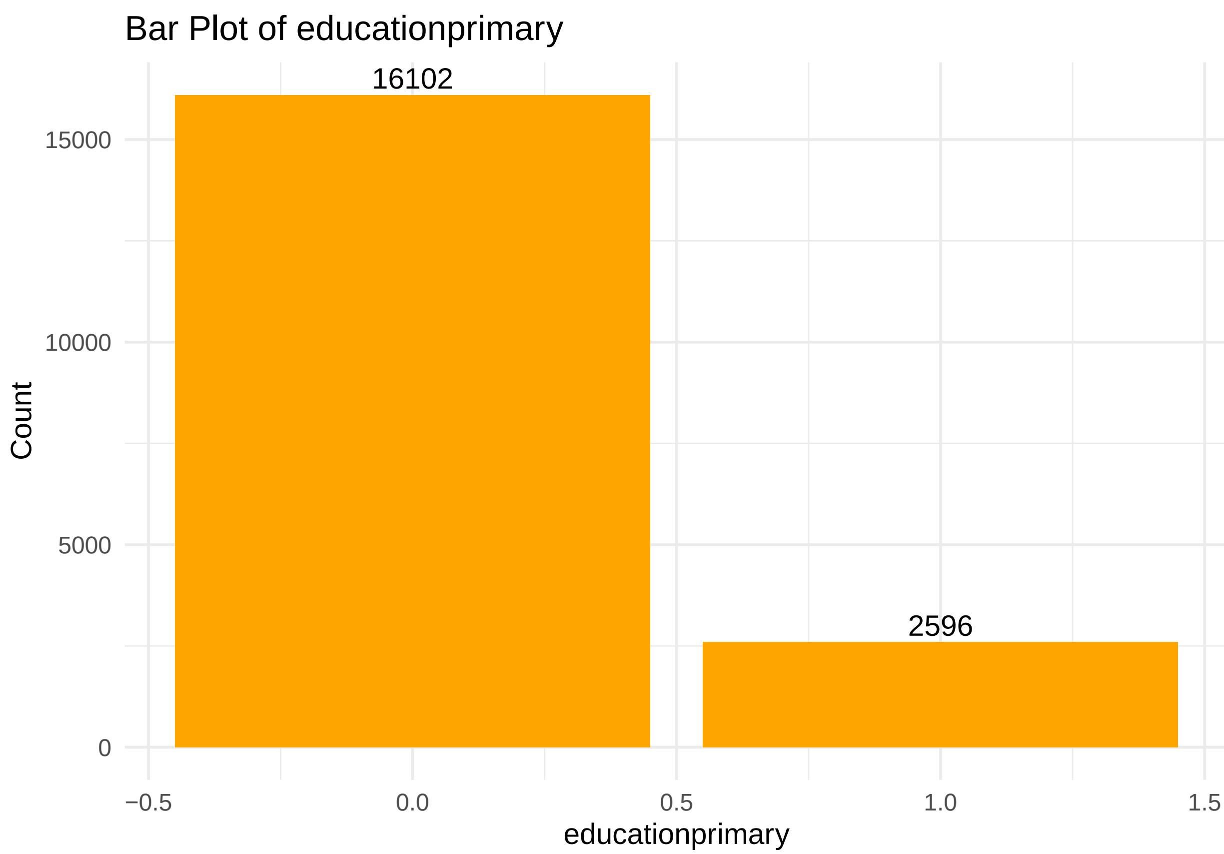


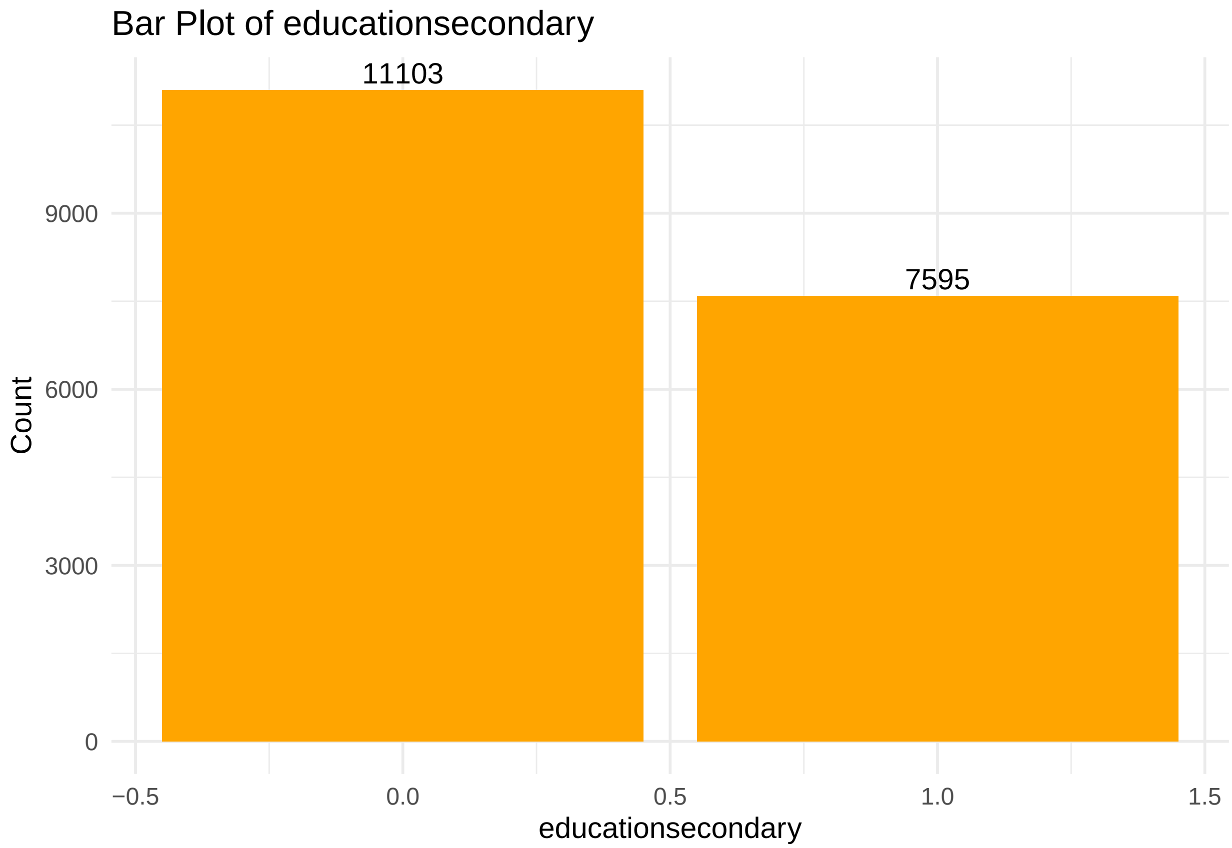
Histogram of LR_scale
Count
4000
2000
0
2.5
5.0
7.5
10.0
LR_scale

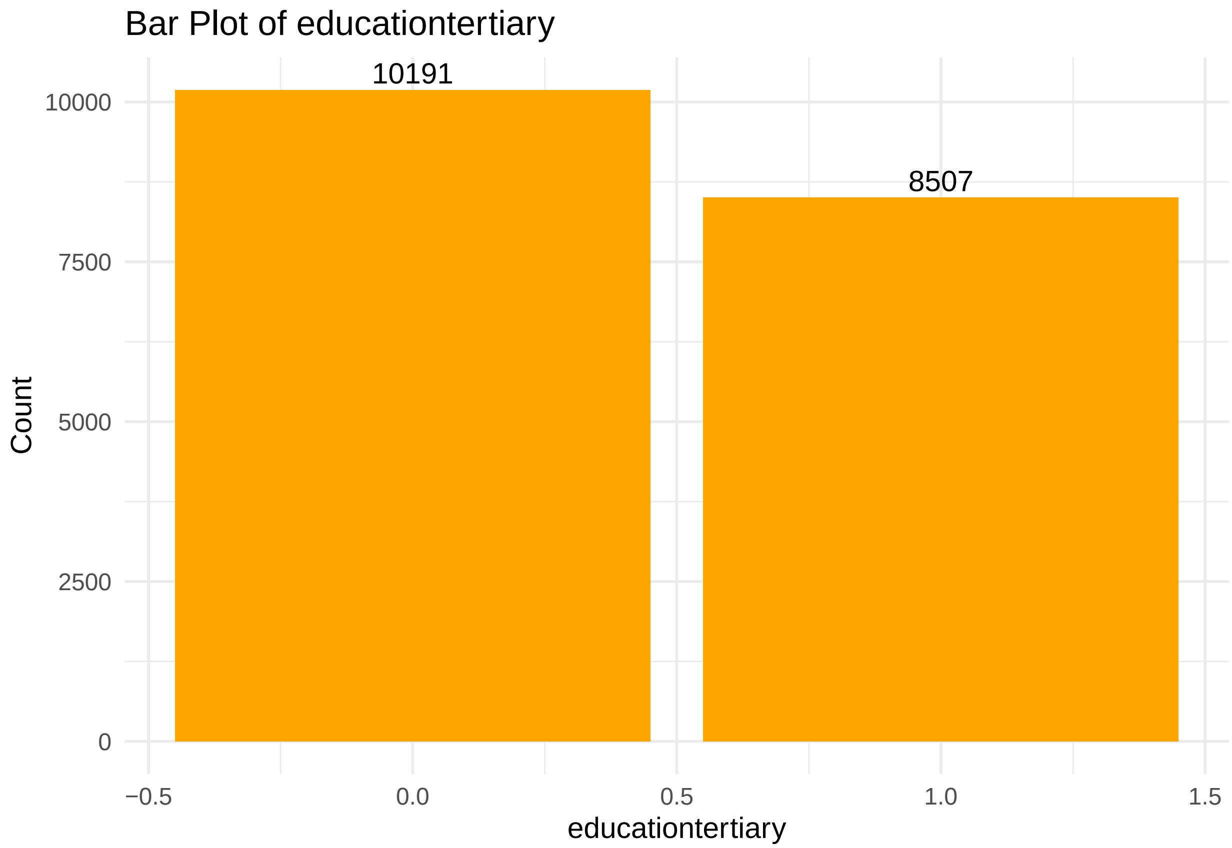

Histogram of new_ccknowledge_index
Count
2000
1000
0
0.25
0.50
0.75
1.00
new_ccknowledge_index

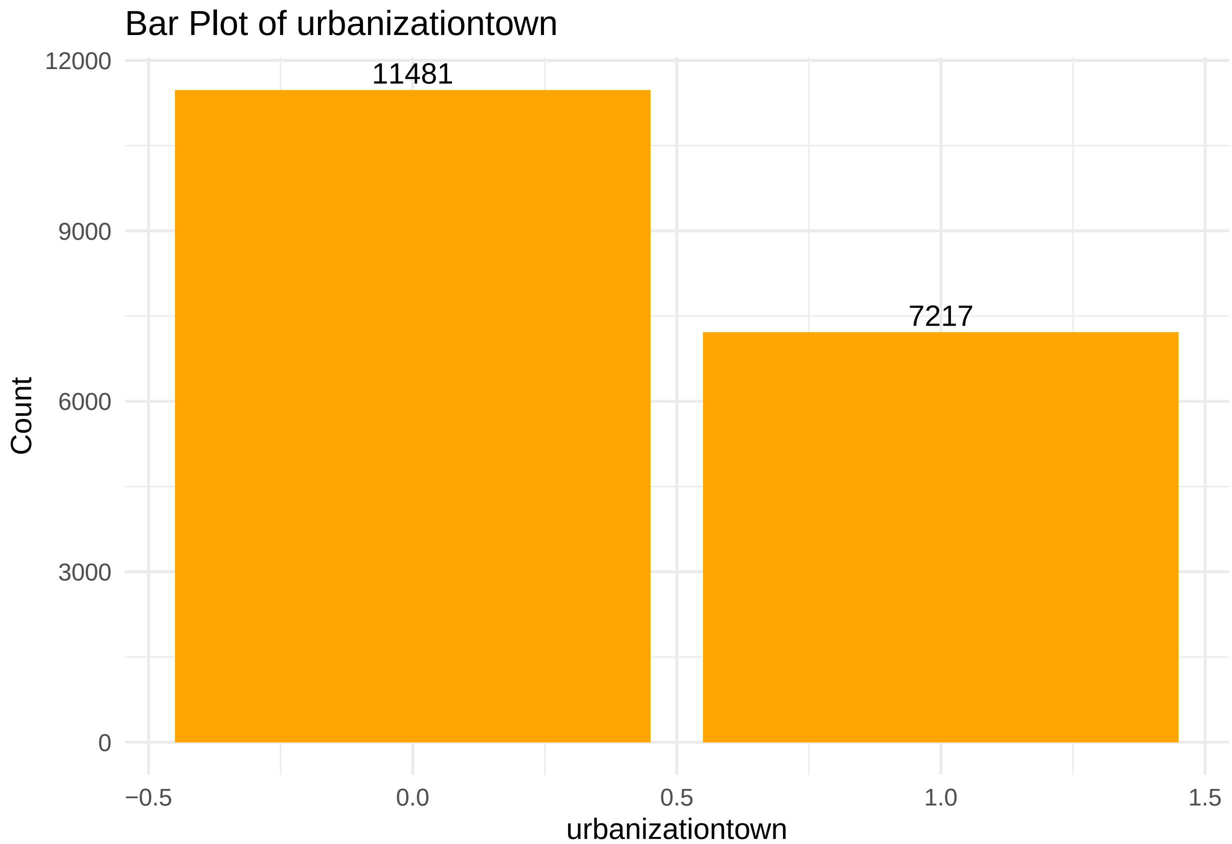

Bar Plot of gender
9244
9454
7500
5000
2500
0
Count
female
male
gender

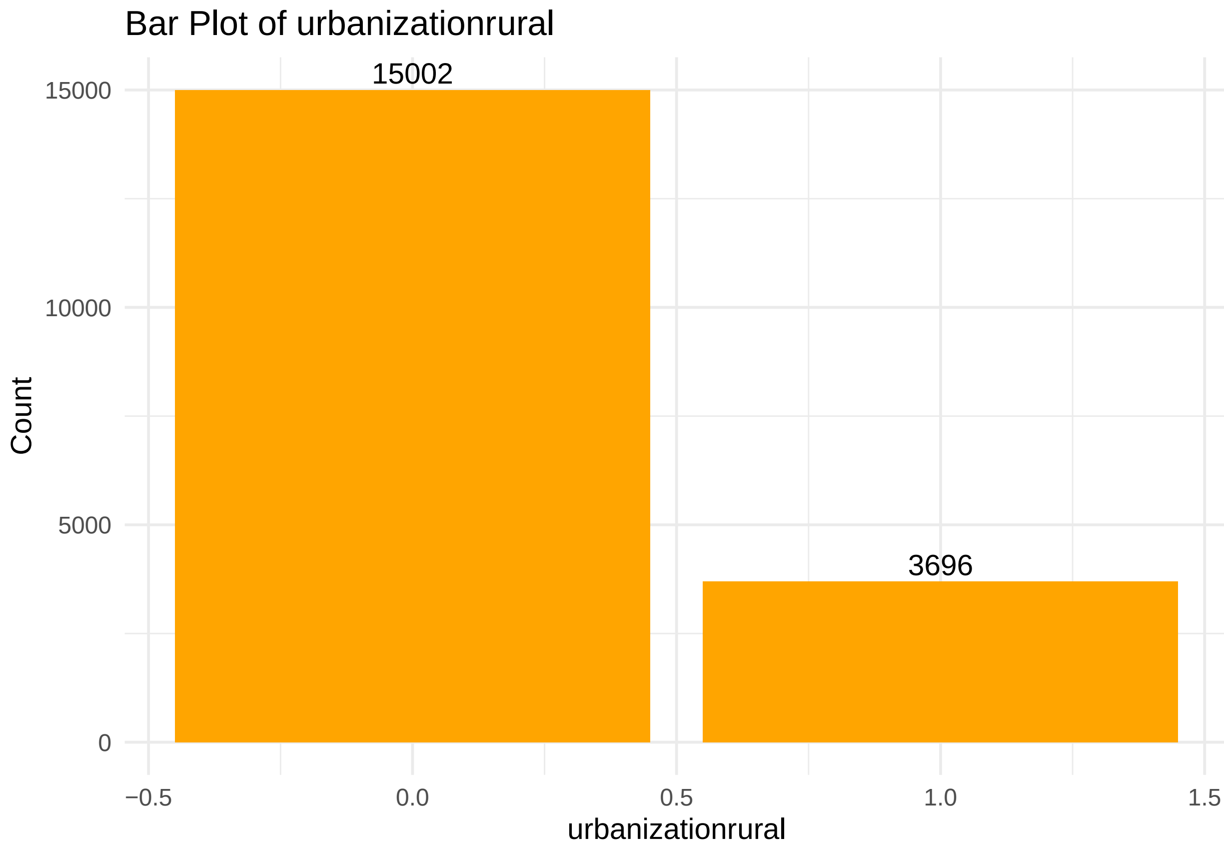

Histogram of country_w
Count
6000
4000
2000
0
0
2
4
6
country_w

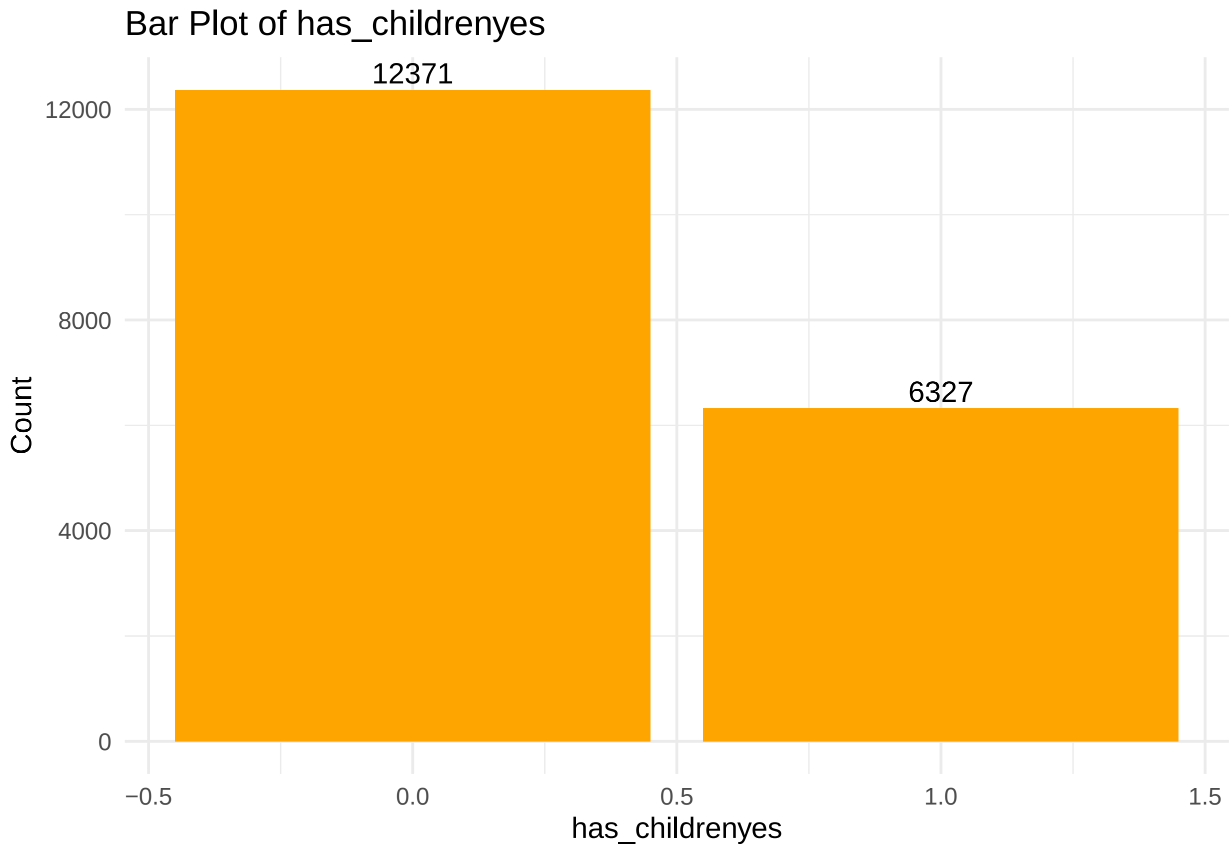

Bar Plot of educationprimary
16102
2596
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
educationprimary

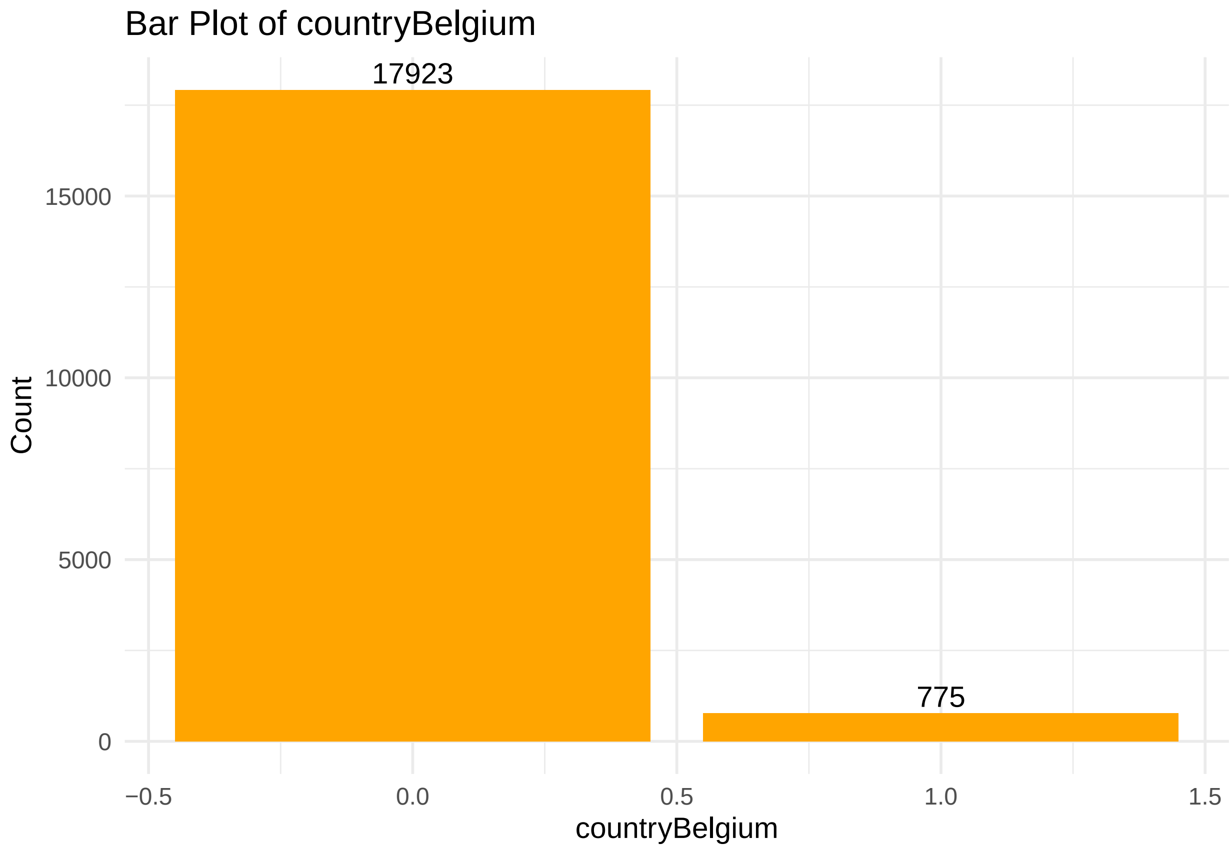

Bar Plot of educationsecondary
11103
7595
9000
6000
3000
0
Count
−0.5
0.0
0.5
1.0
1.5
educationsecondary

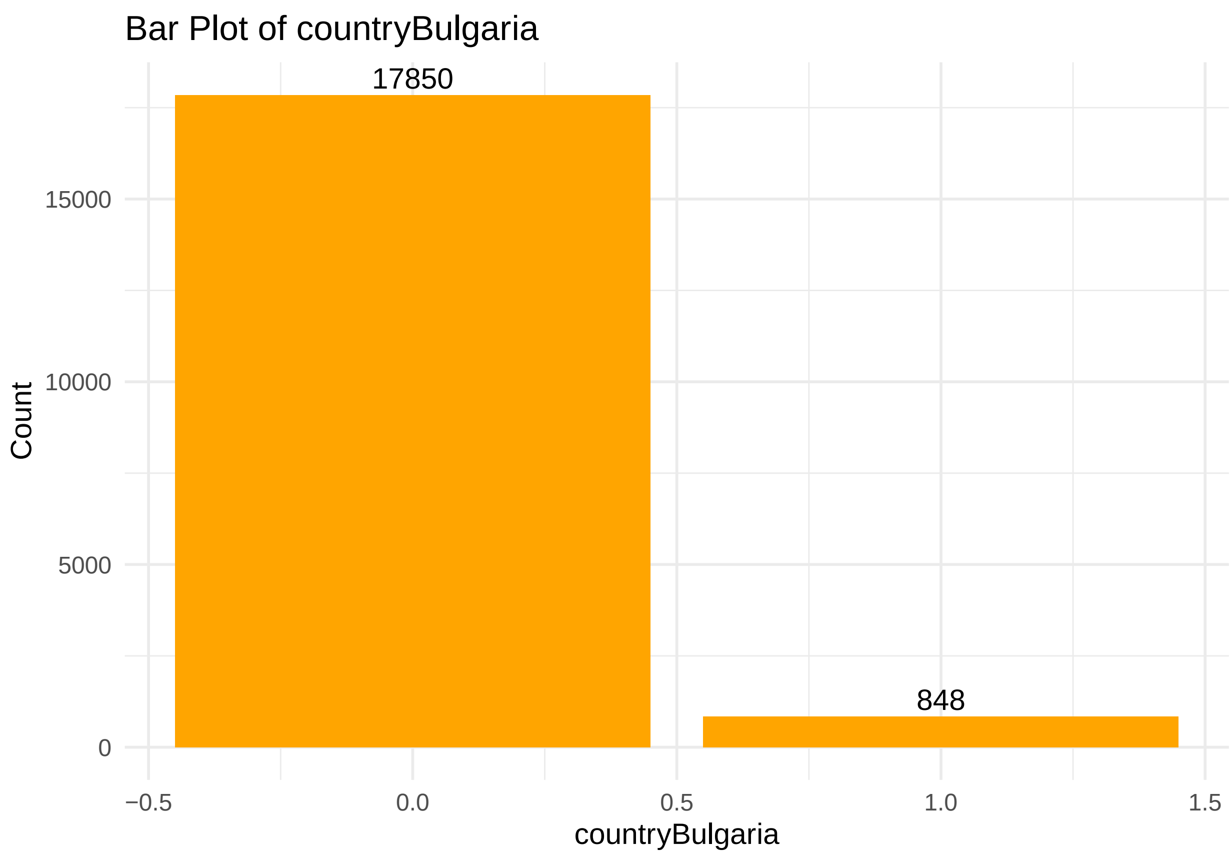

Bar Plot of educationtertiary
10191
8507
10000
7500
5000
2500
0
Count
−0.5
0.0
0.5
1.0
1.5
educationtertiary

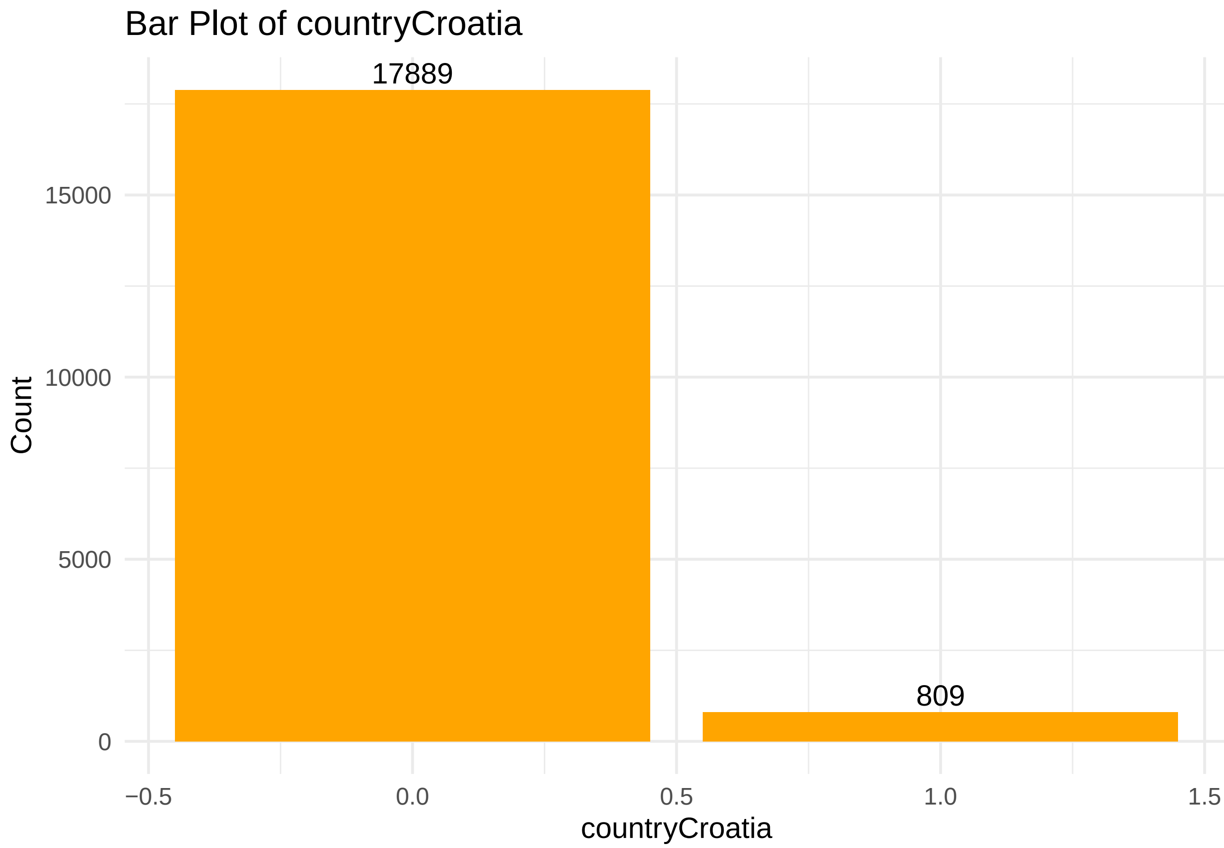

Bar Plot of urbanizationtown
12000
9000
6000
3000
0
Count
11481
7217
−0.5
0.0
0.5
1.0
1.5
urbanizationtown


Bar Plot of urbanizationrural
15002
3696
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
urbanizationrural

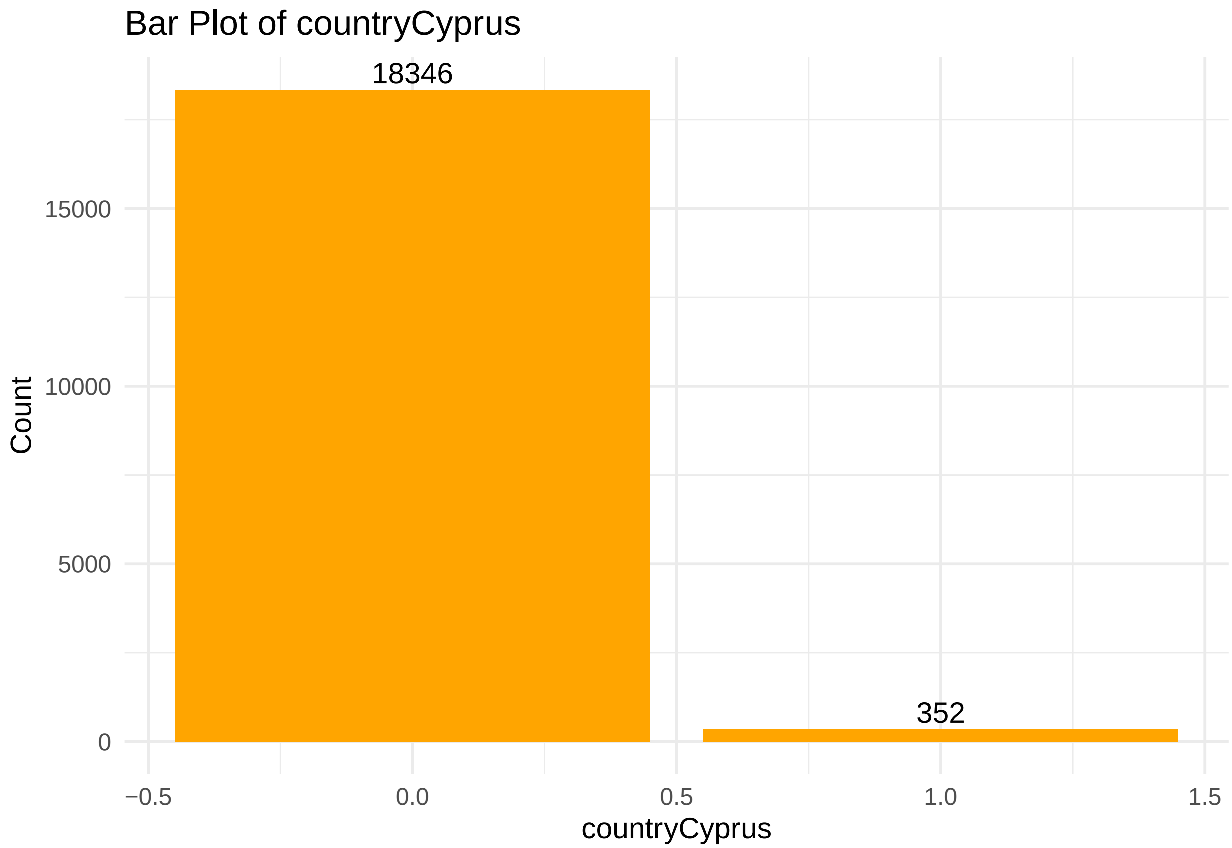

Bar Plot of has_childrenyes
12371
6327
12000
8000
4000
0
Count
−0.5
0.0
0.5
1.0
1.5
has_childrenyes

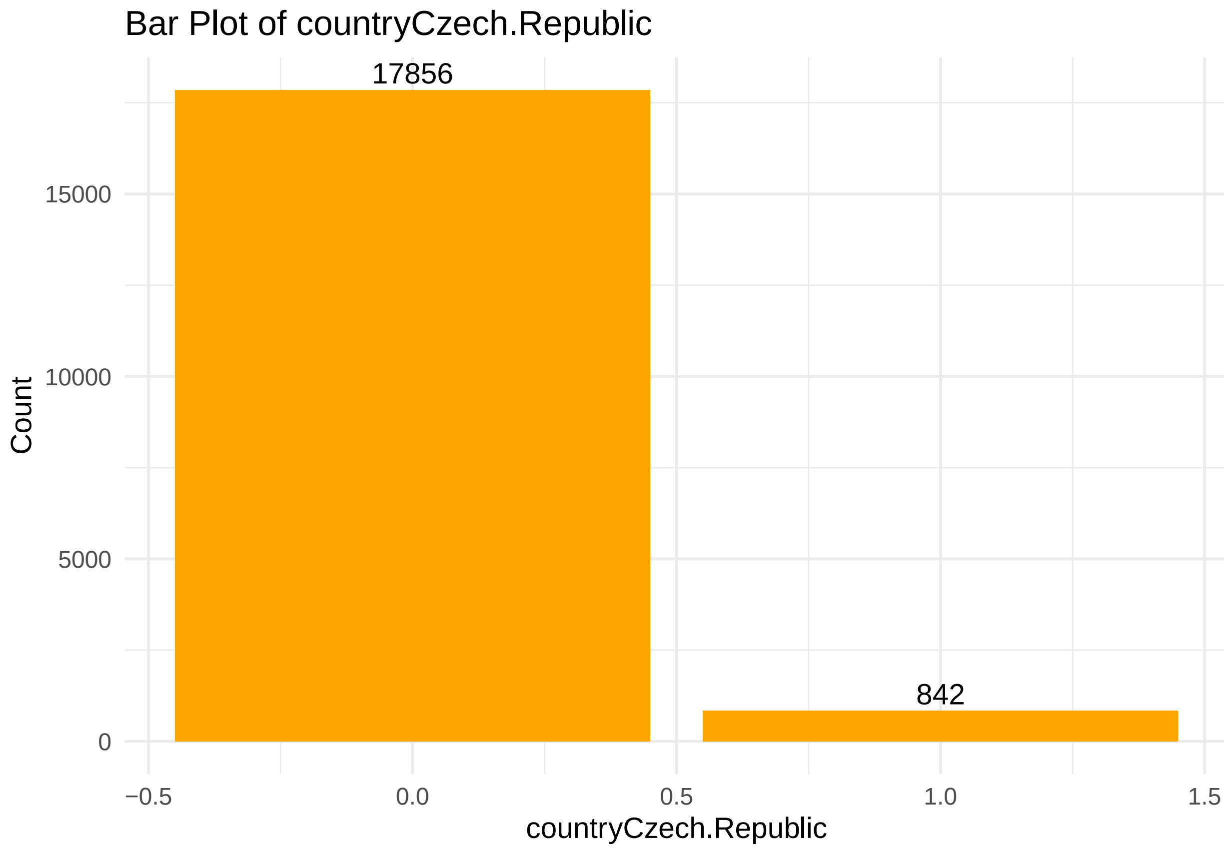

Bar Plot of countryBelgium
17923
775
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryBelgium

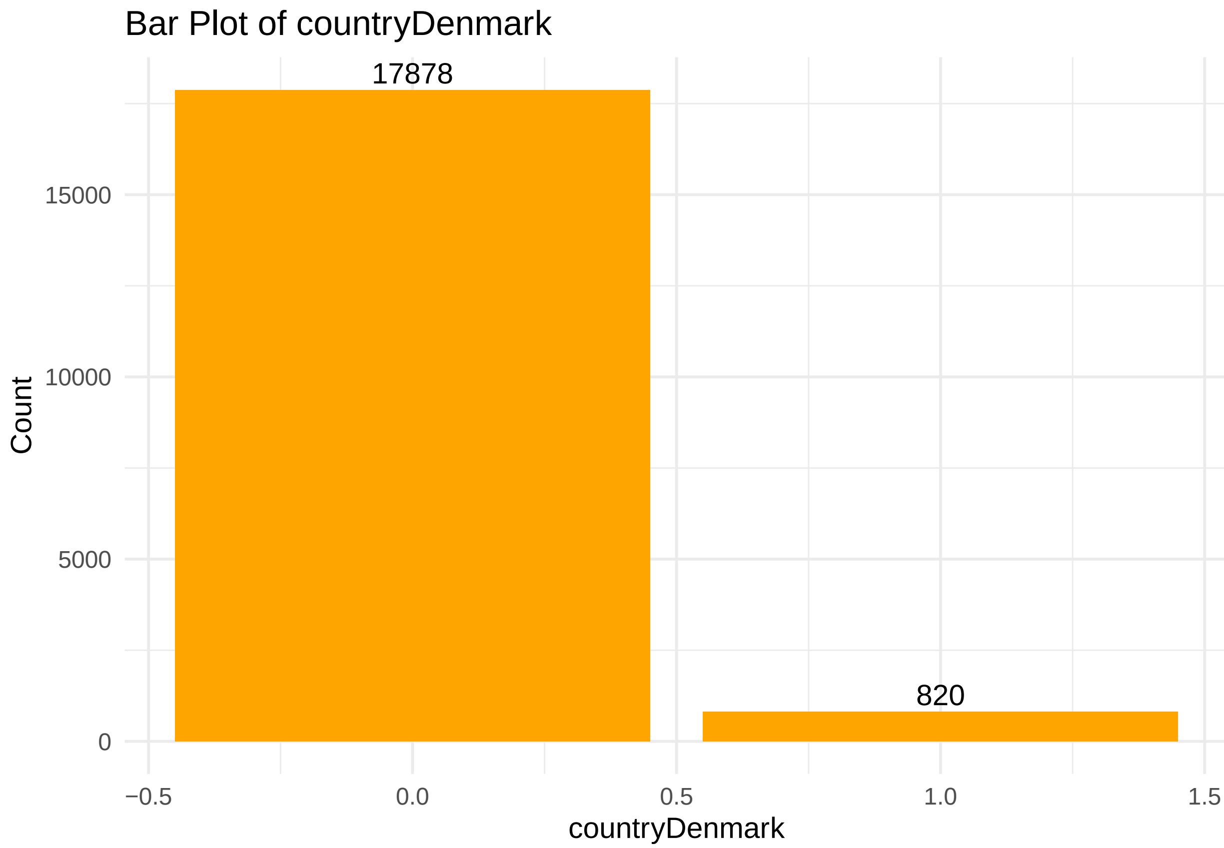

Bar Plot of countryBulgaria
17850
848
Count
15000
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryBulgaria

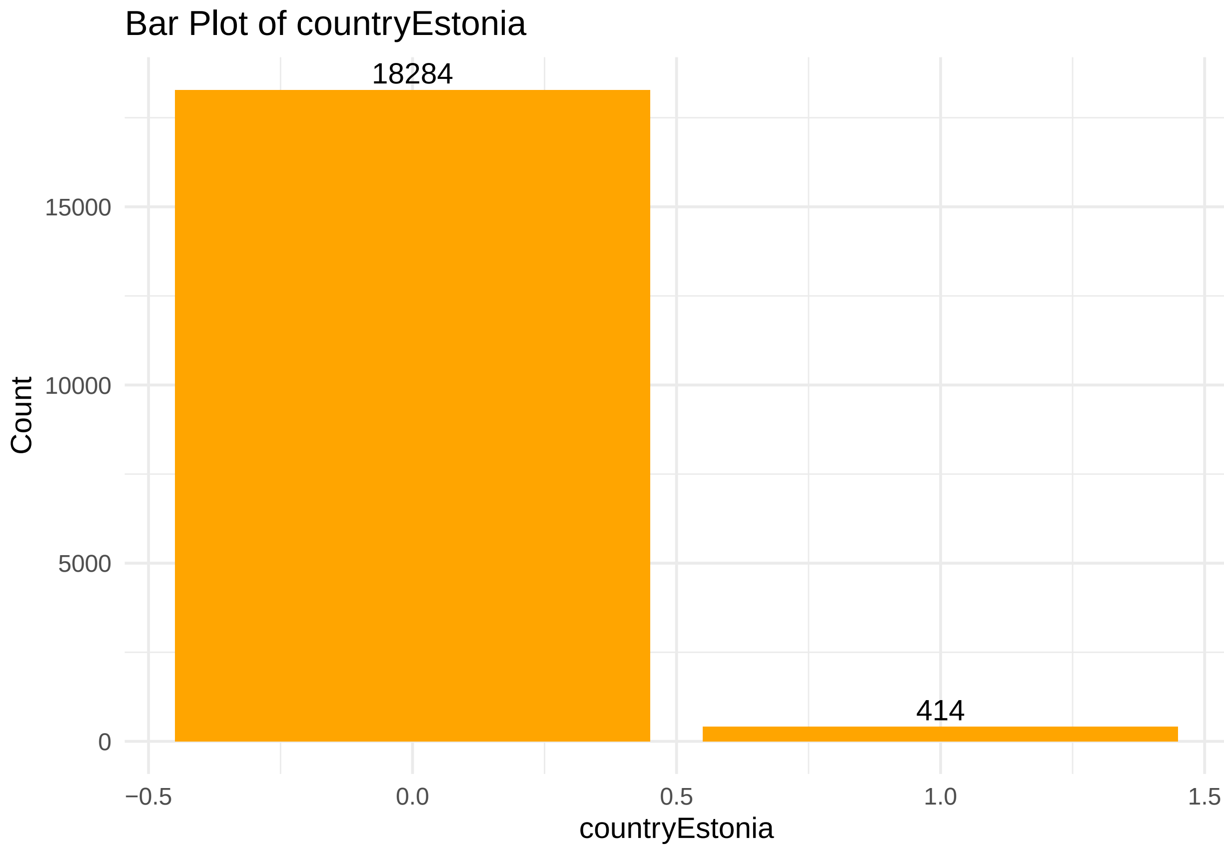


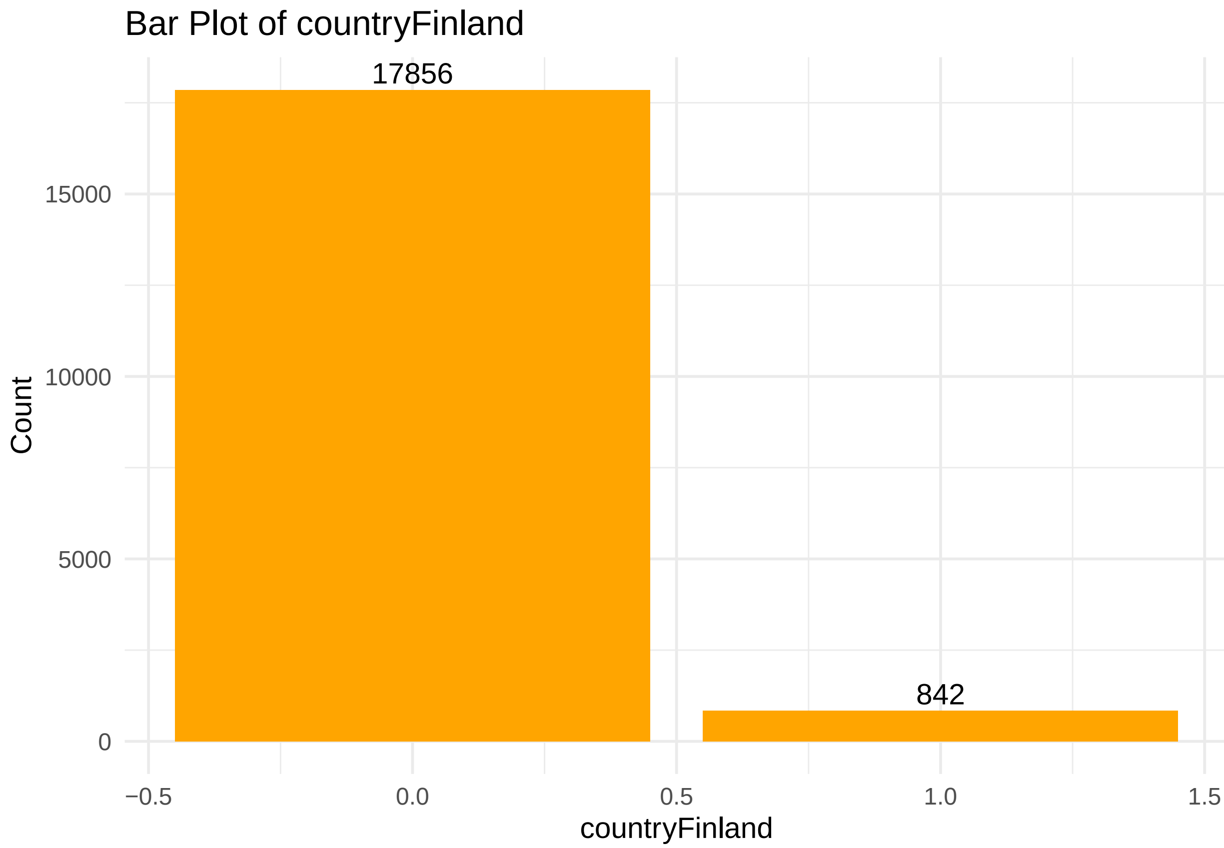
Bar Plot of countryCroatia
17889
809
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryCroatia

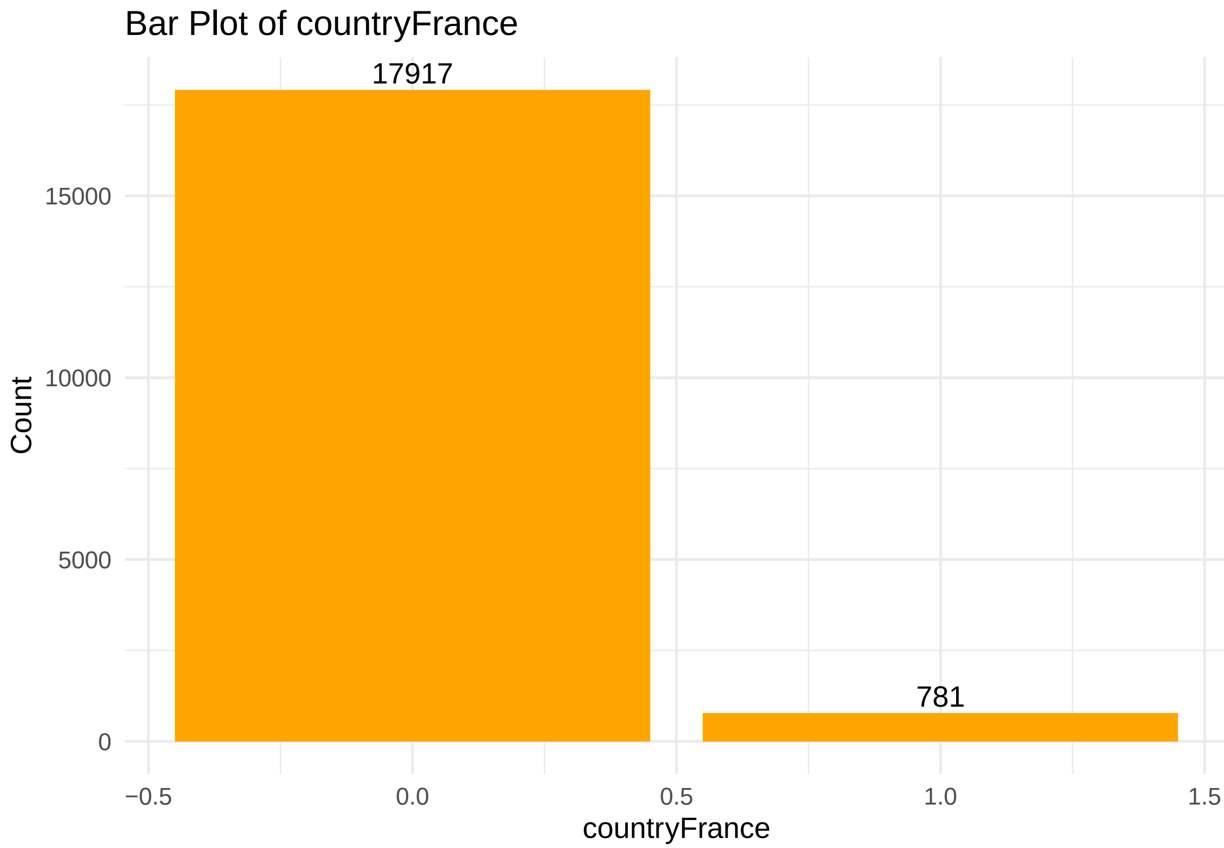

Bar Plot of countryCyprus
18346
352
Count
15000
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryCyprus

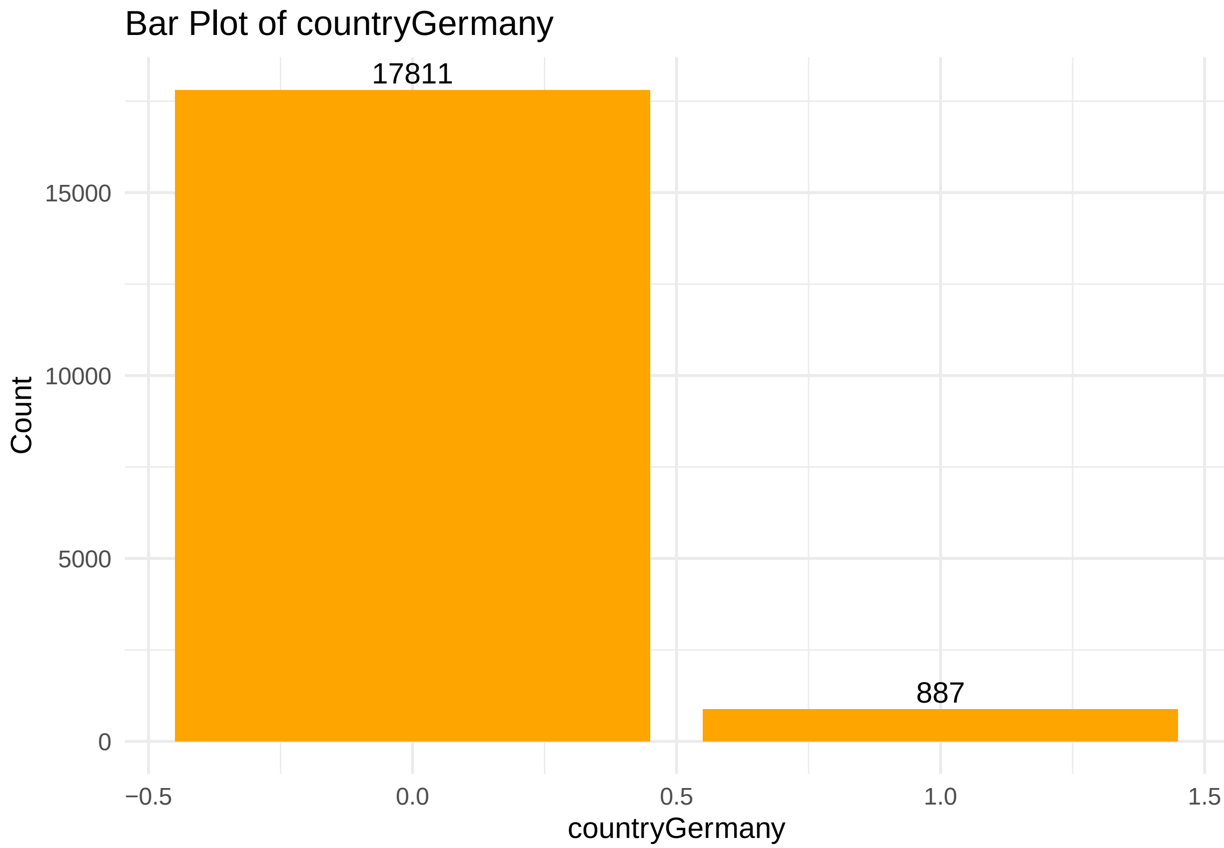

Bar Plot of countryCzech.Republic
17856
842
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryCzech.Republic


Bar Plot of countryDenmark
17878
820
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryDenmark

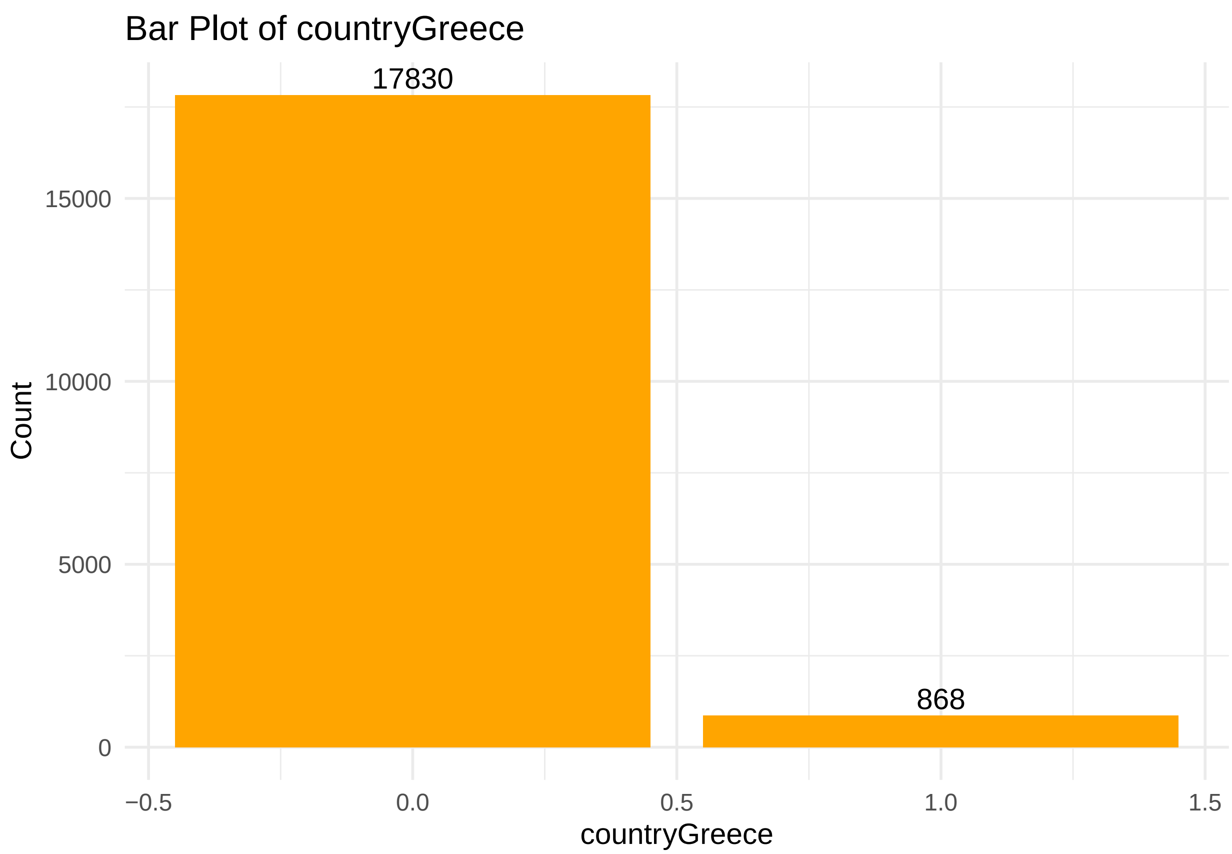

Bar Plot of countryEstonia
18284
414
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryEstonia


Bar Plot of countryFinland
17856
842
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryFinland

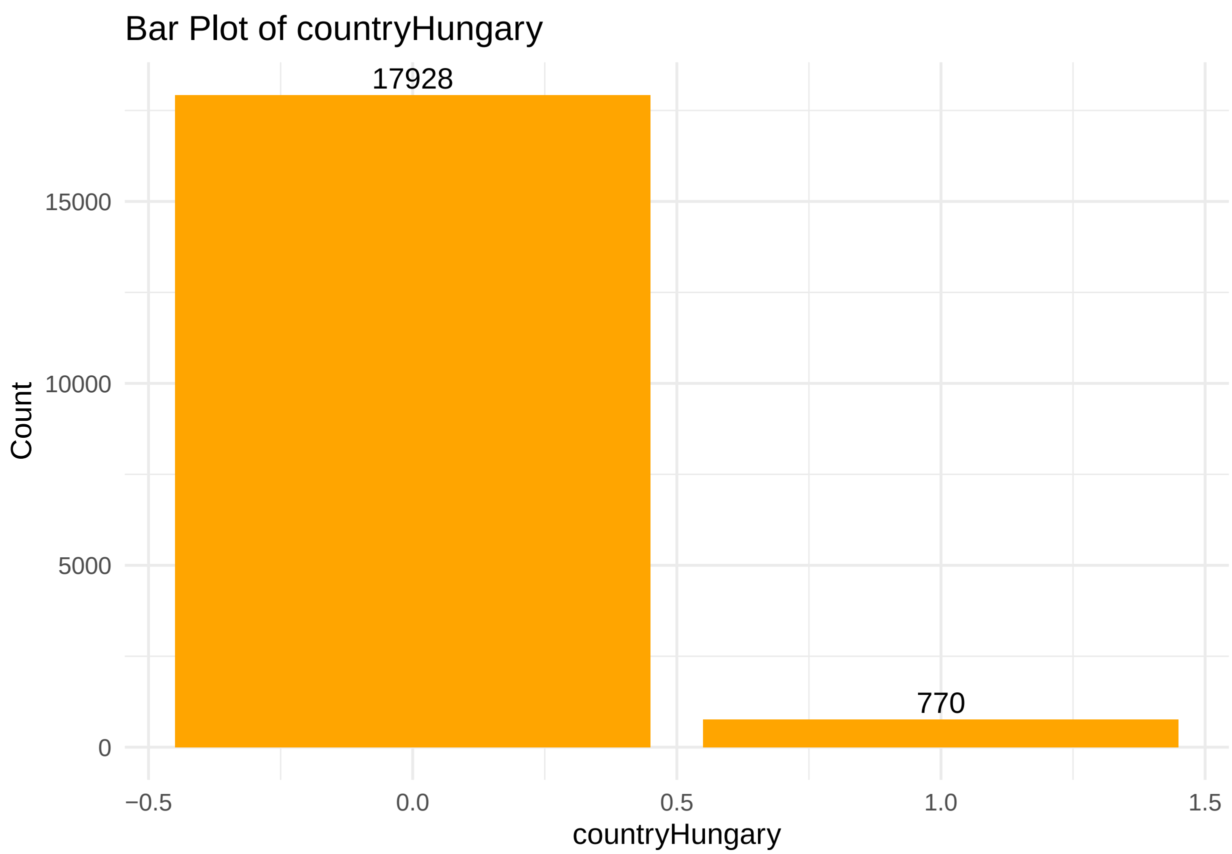

Bar Plot of countryFrance
17917
781
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryFrance

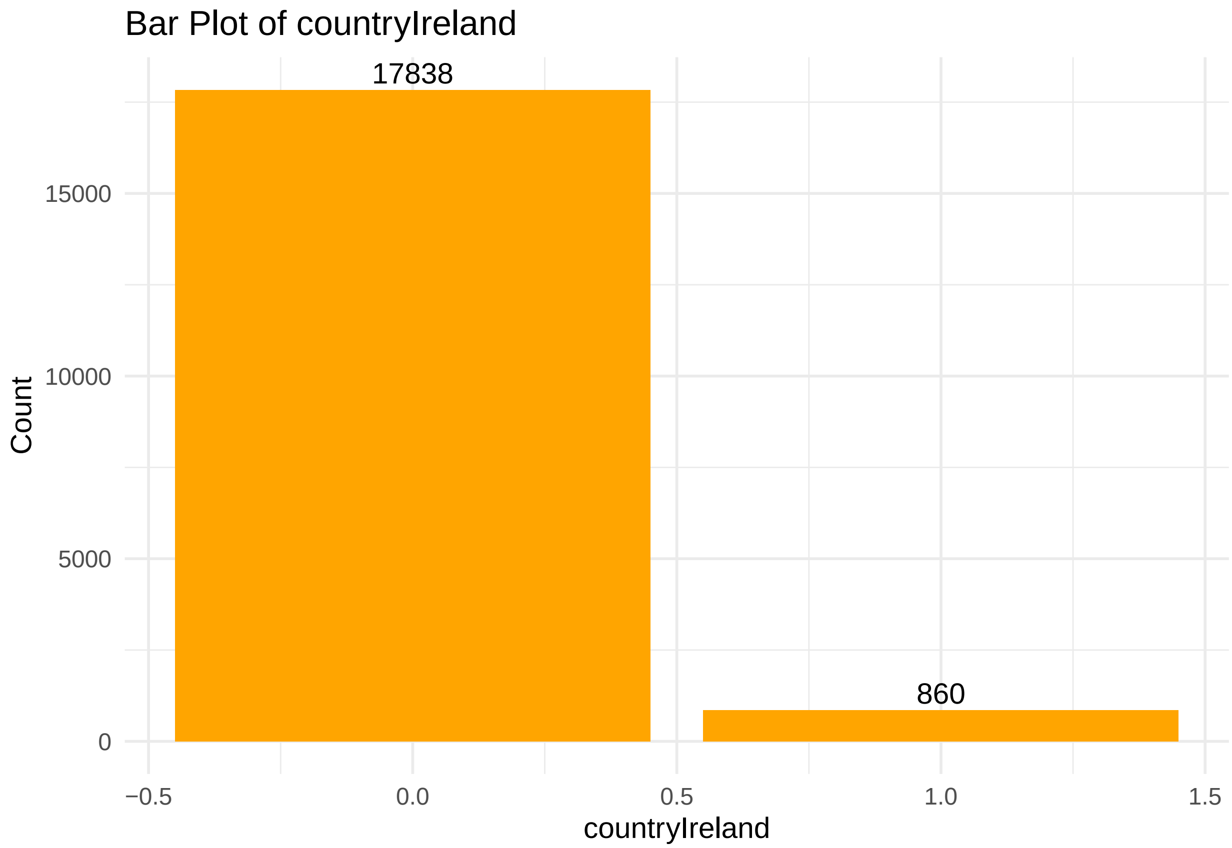


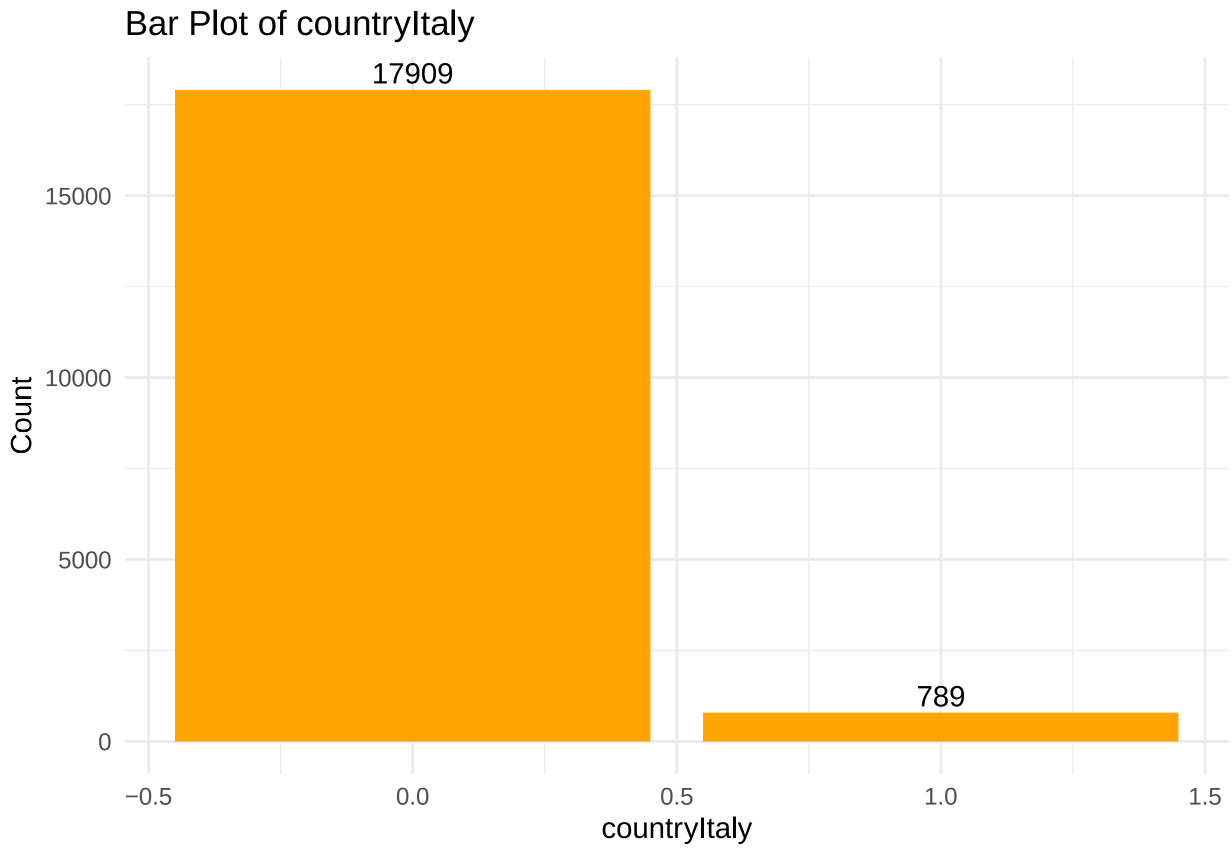
Bar Plot of countryGermany
17811
887
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryGermany

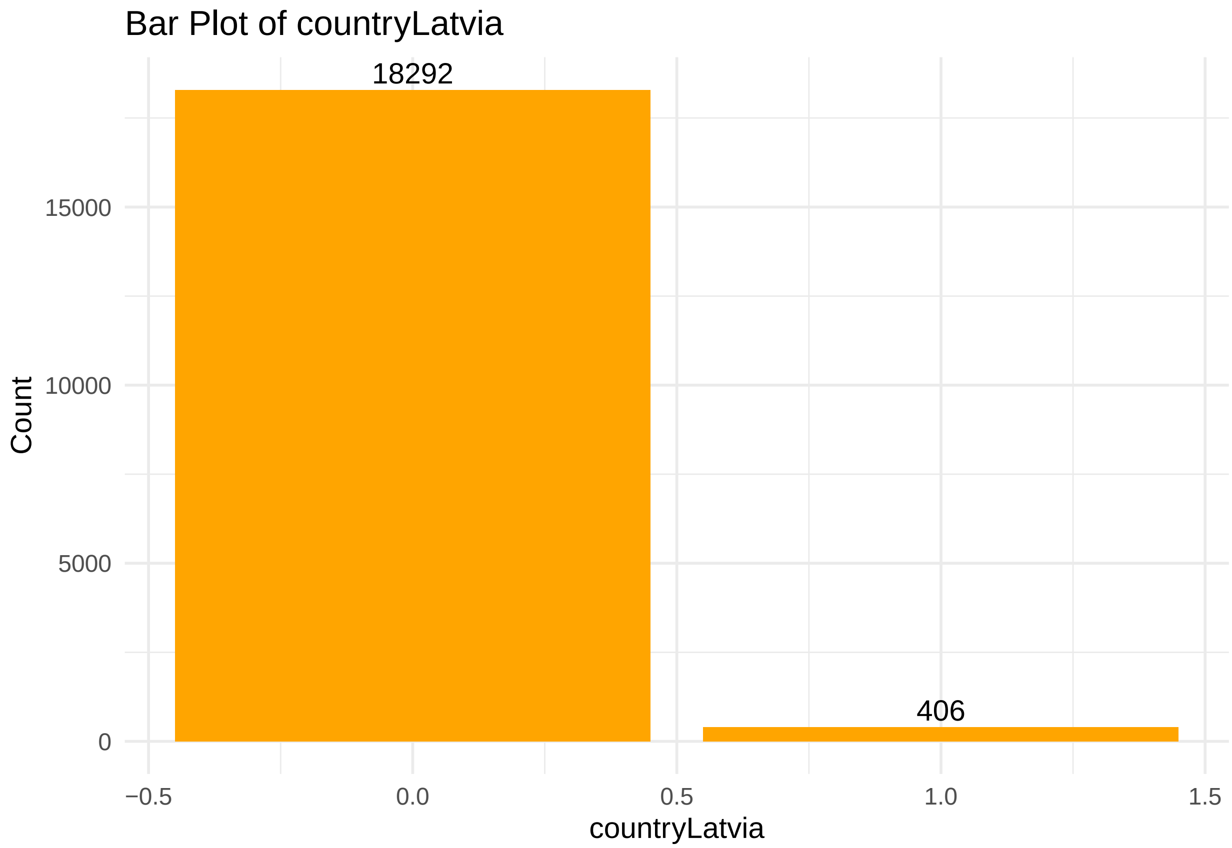


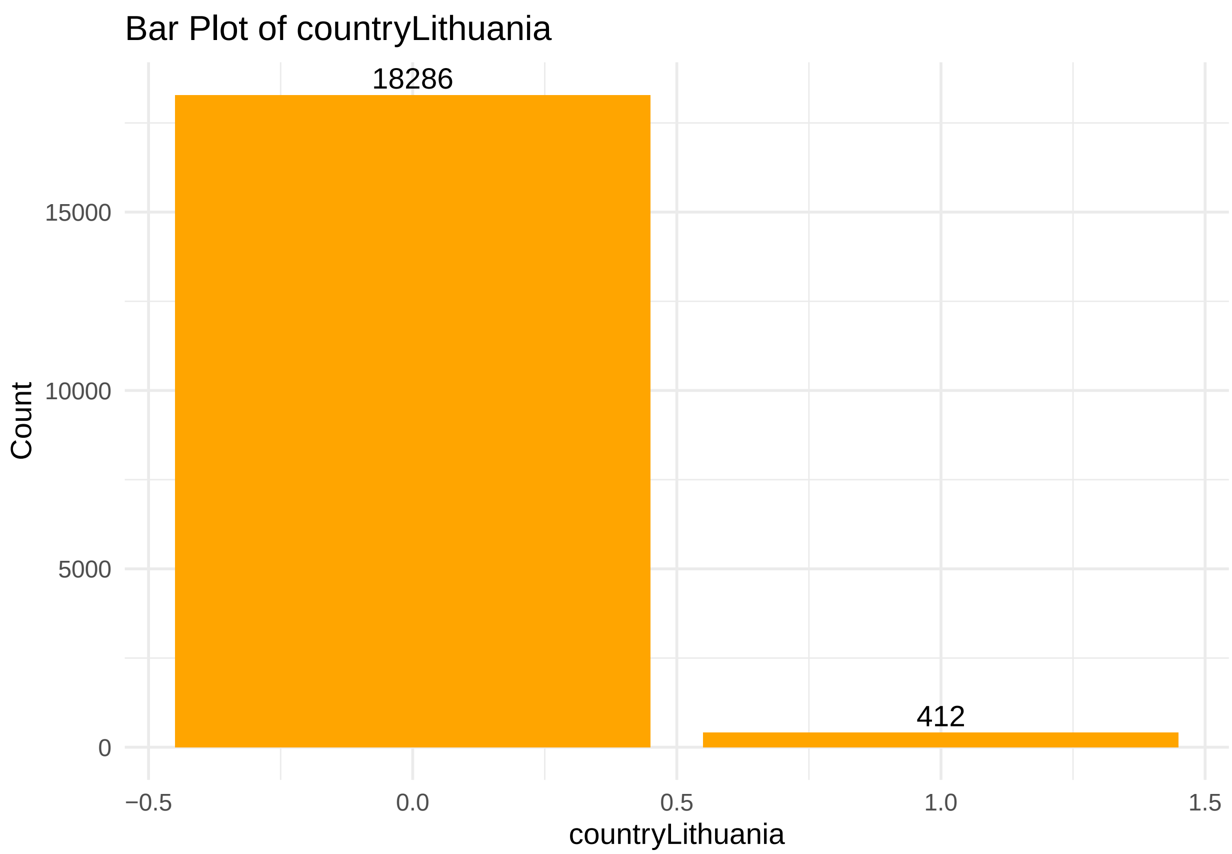
Bar Plot of countryGreece
17830
868
Count
15000
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryGreece

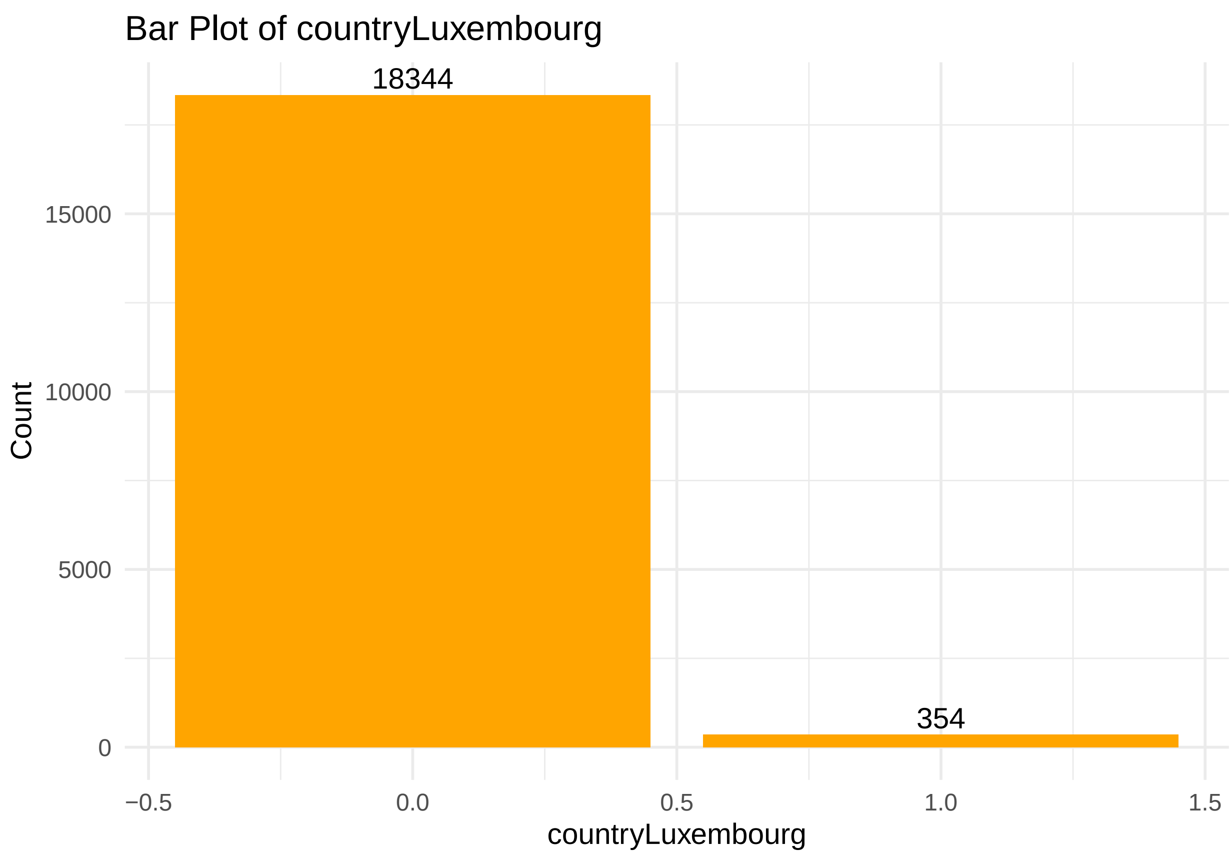

Bar Plot of countryHungary
17928
770
Count
15000
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryHungary

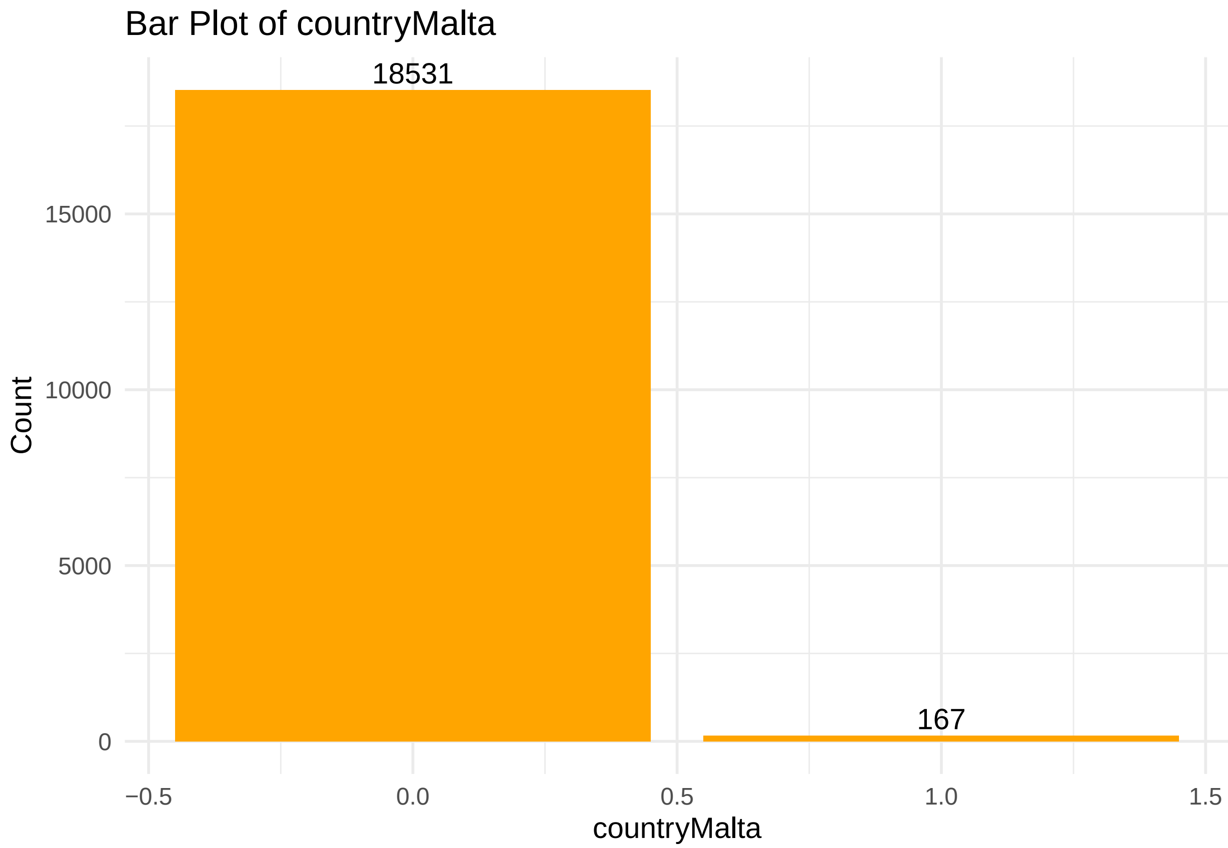

Bar Plot of countryIreland
17838
860
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryIreland


Bar Plot of countryItaly
17909
789
Count
15000
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryItaly

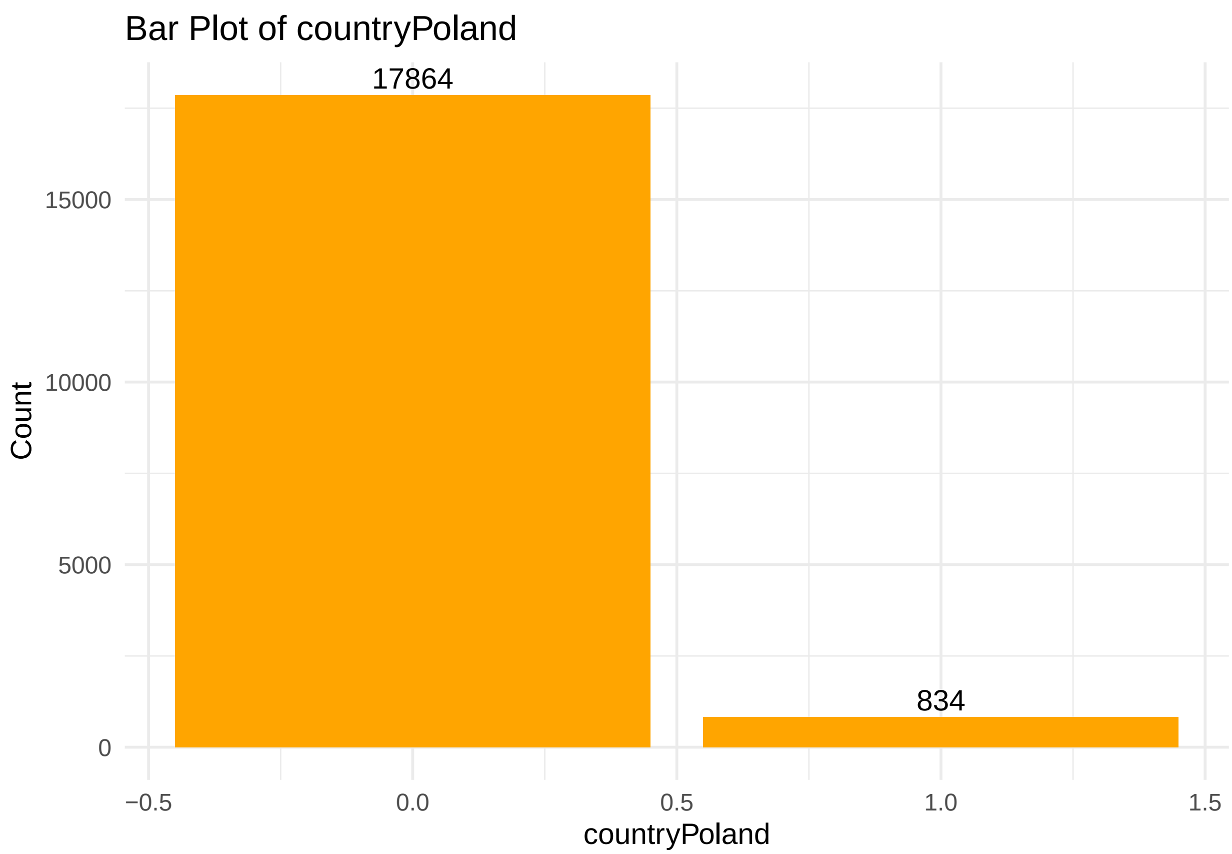

Bar Plot of countryLatvia
18292
406
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryLatvia


Bar Plot of countryLithuania
18286
412
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryLithuania

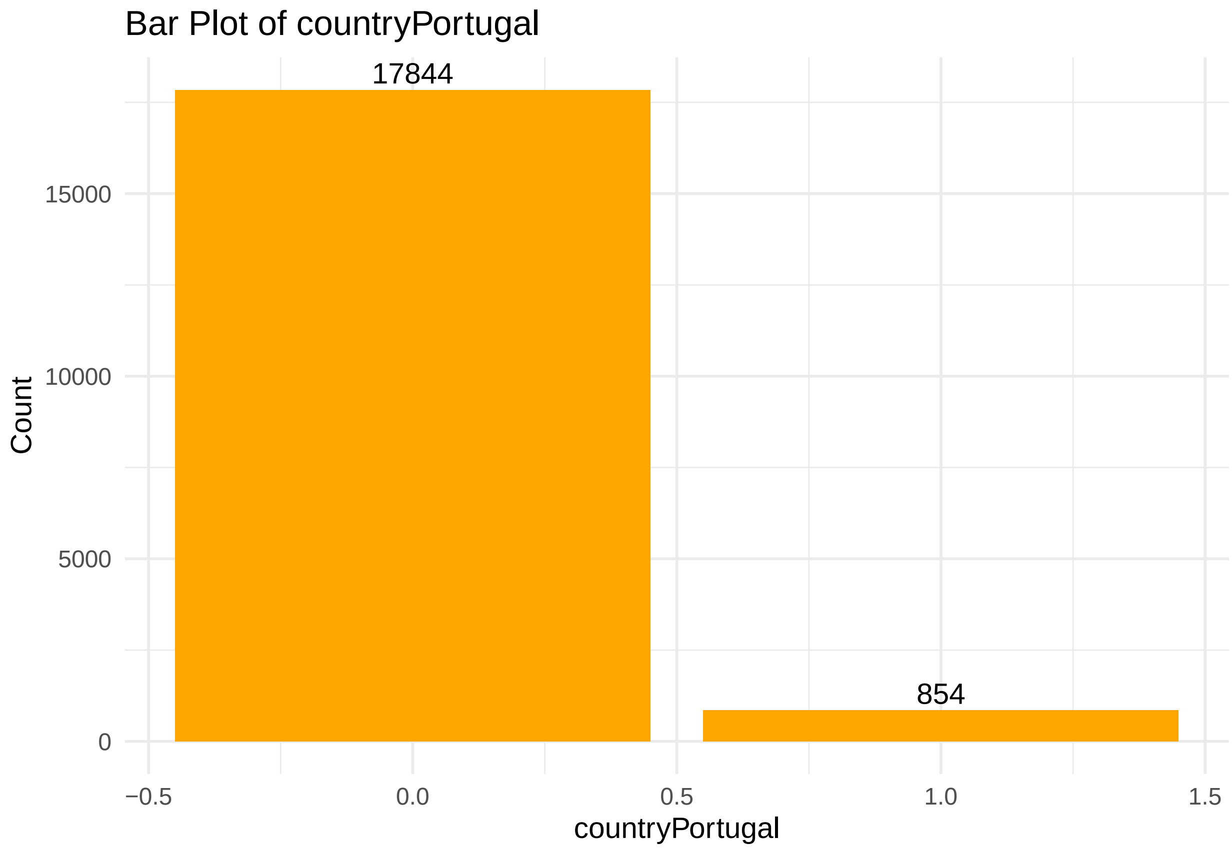

Bar Plot of countryLuxembourg
18344
354
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryLuxembourg

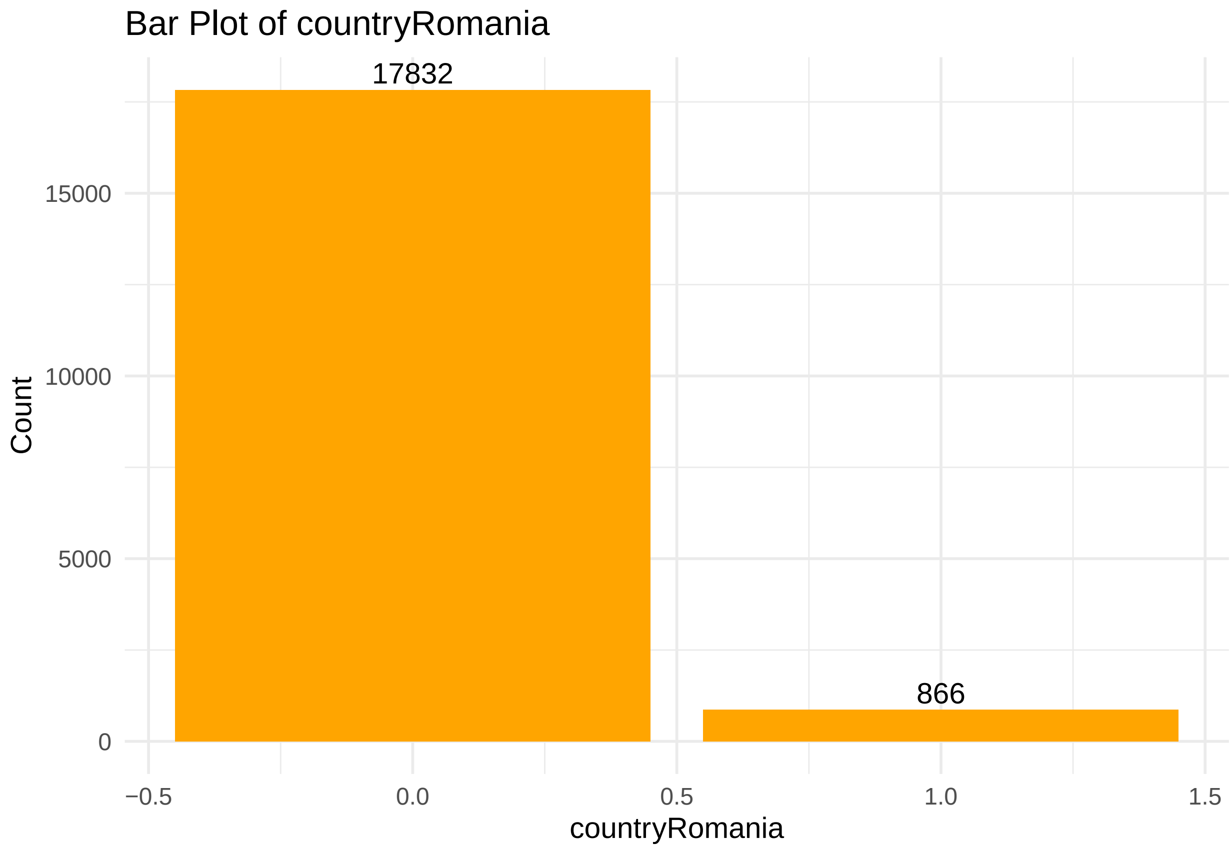


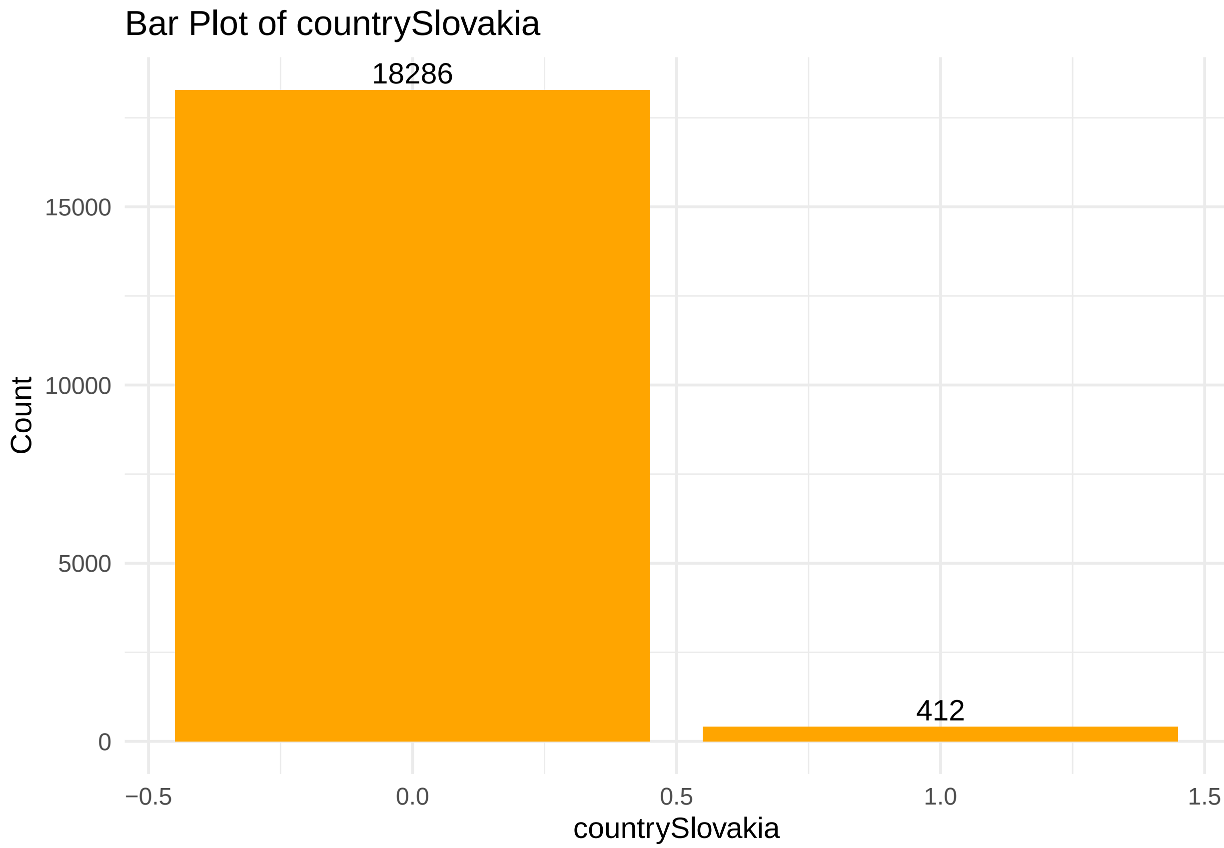
Bar Plot of countryMalta
18531
167
Count
15000
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryMalta



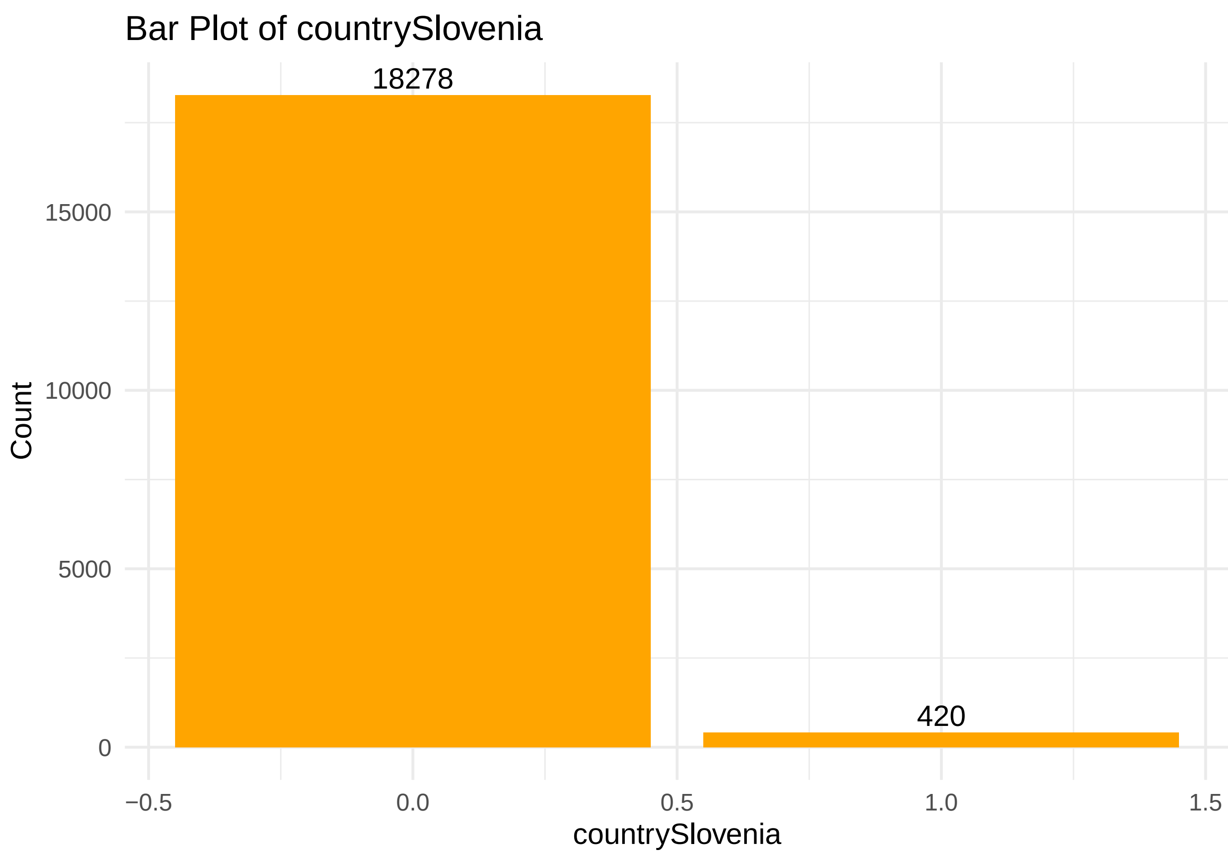
Bar Plot of countryPoland
17864
834
Count
15000
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryPoland

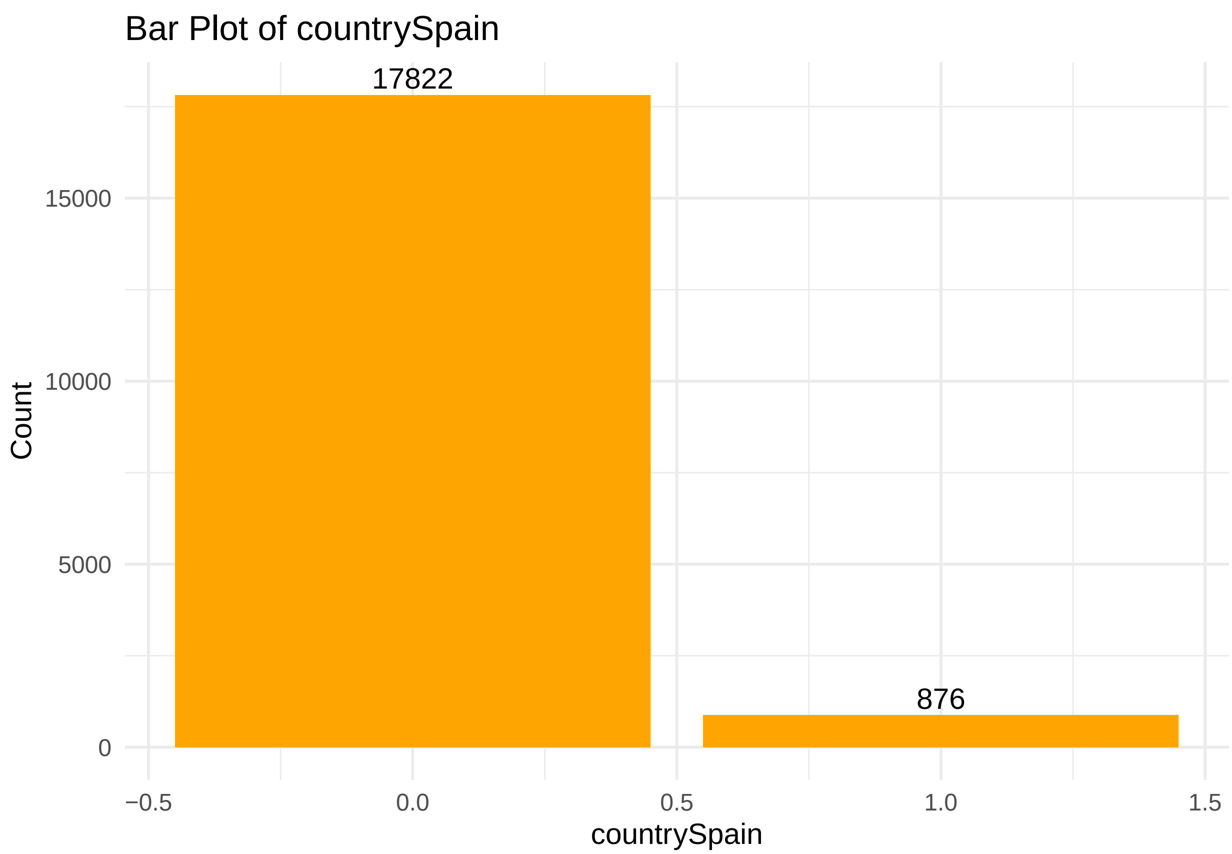

Bar Plot of countryPortugal
17844
854
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryPortugal


Bar Plot of countryRomania
17832
866
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryRomania

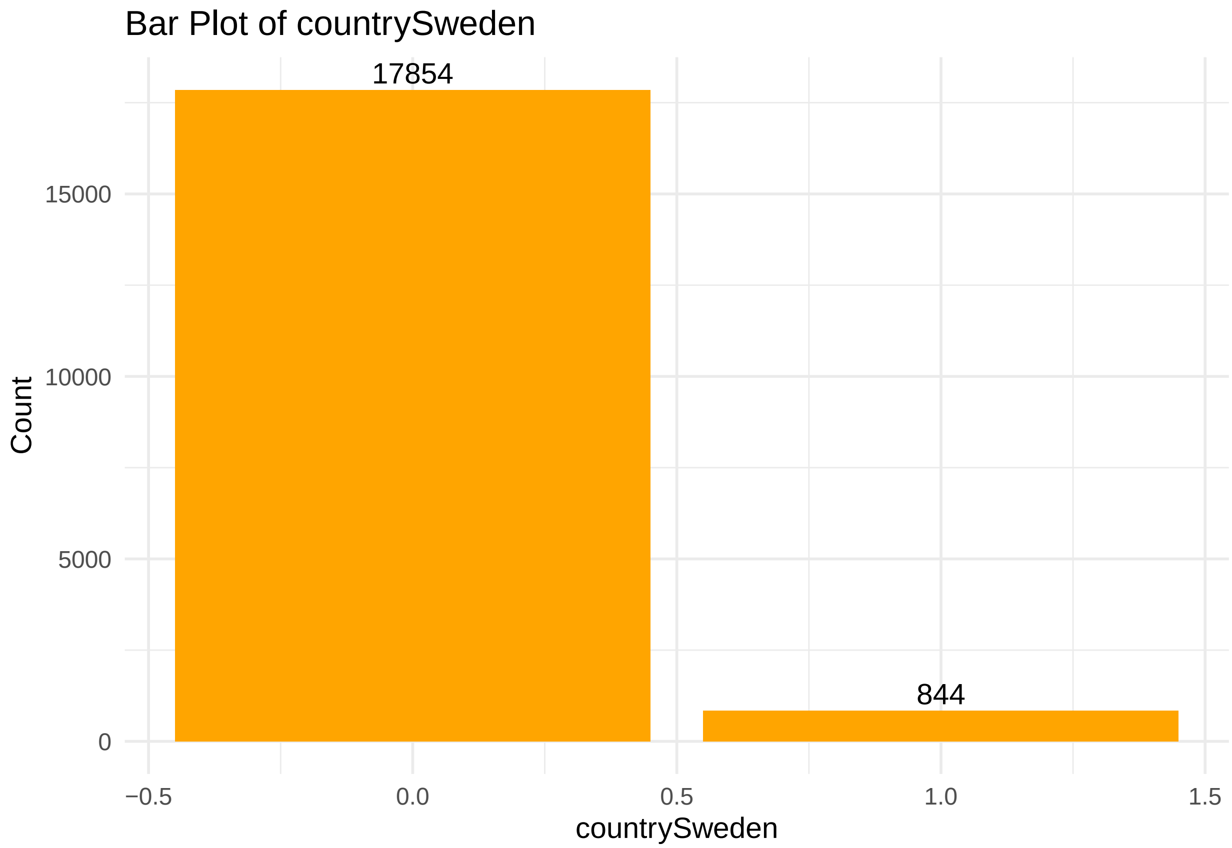


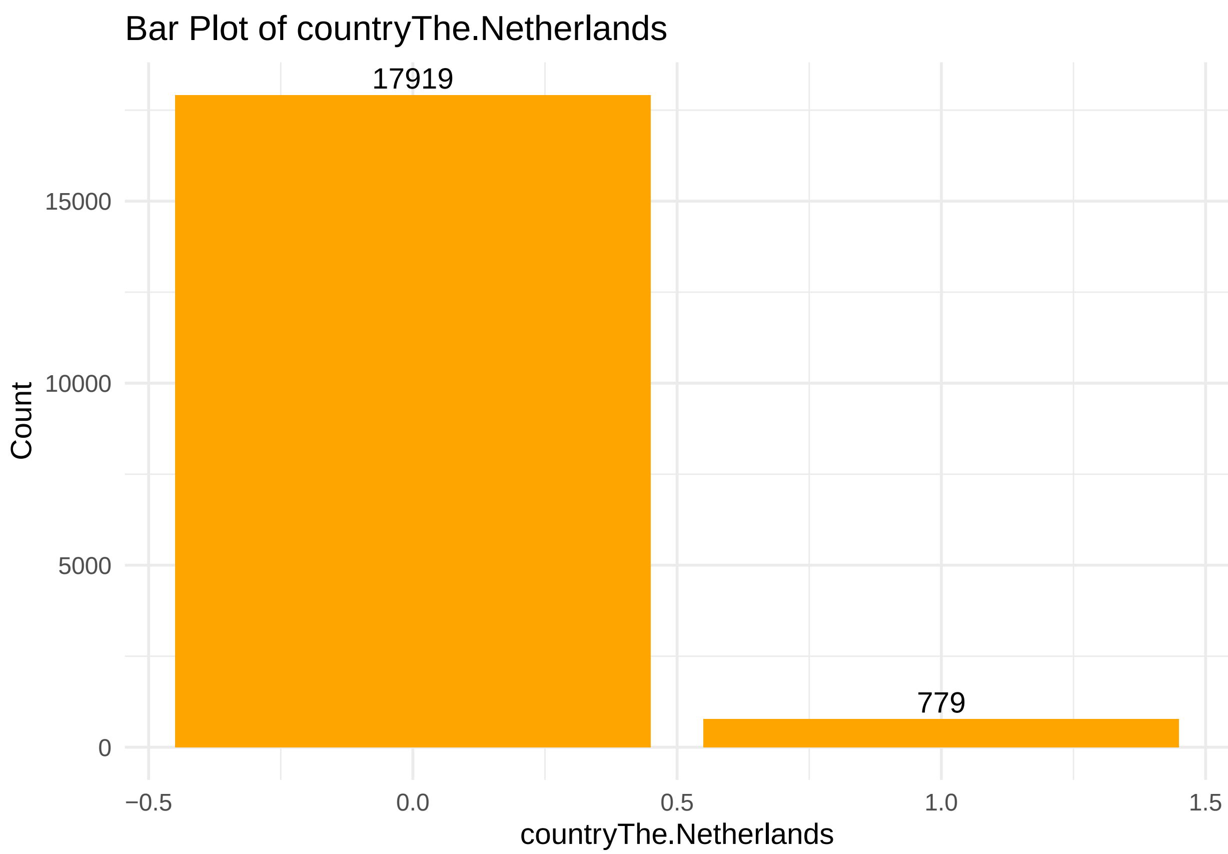
Bar Plot of countrySlovakia
18286
412
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countrySlovakia


Bar Plot of countrySlovenia
18278
420
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countrySlovenia



Bar Plot of countrySpain
17822
876
Count
15000
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countrySpain


Bar Plot of countrySweden
17854
844
15000
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countrySweden

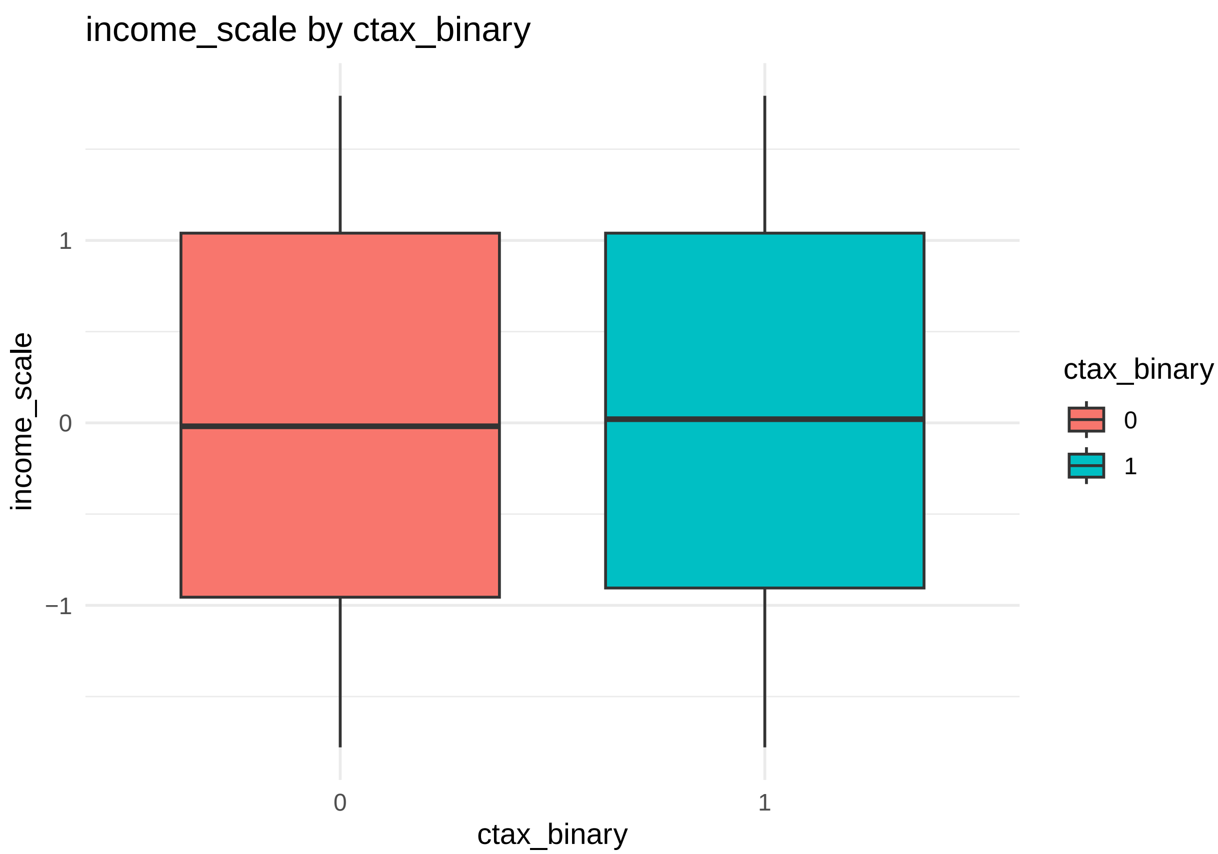


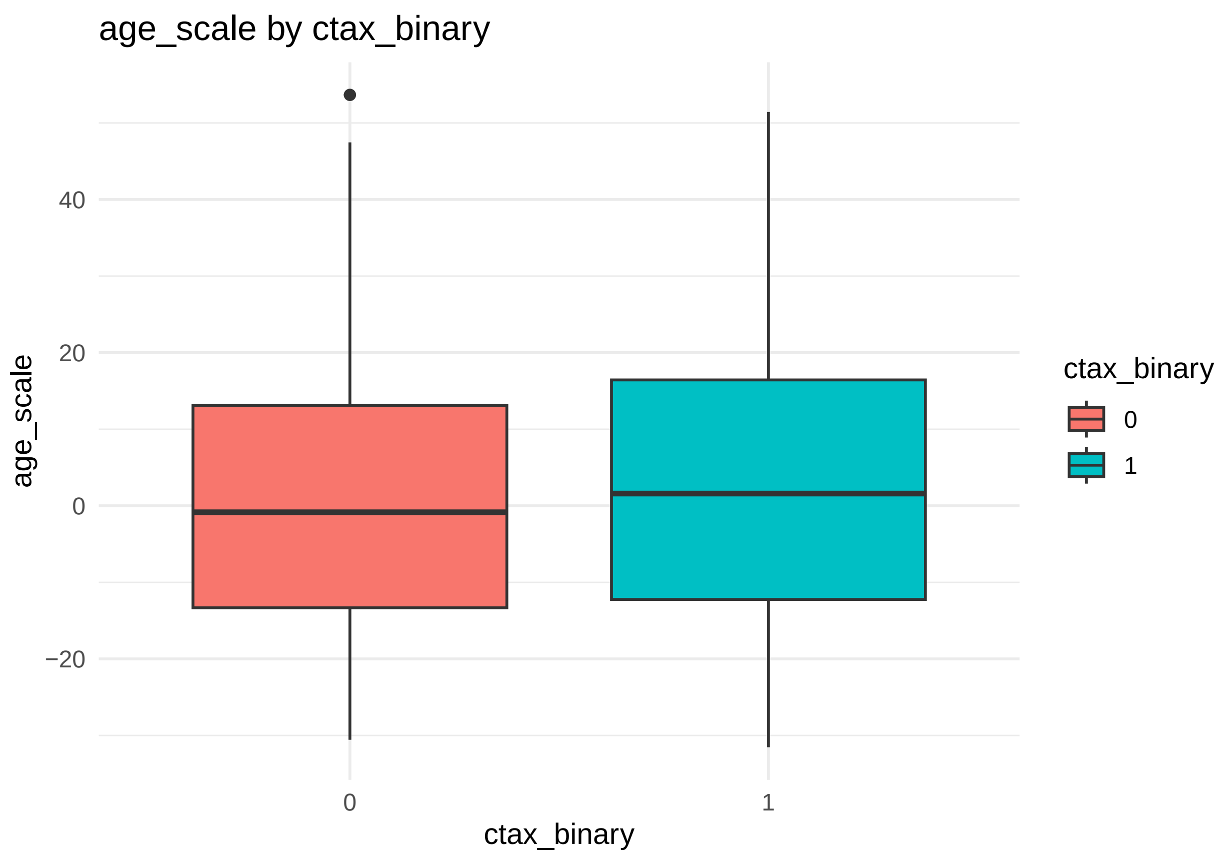
What are the distributions of the independent variables respect to ctax_binary (response) ?

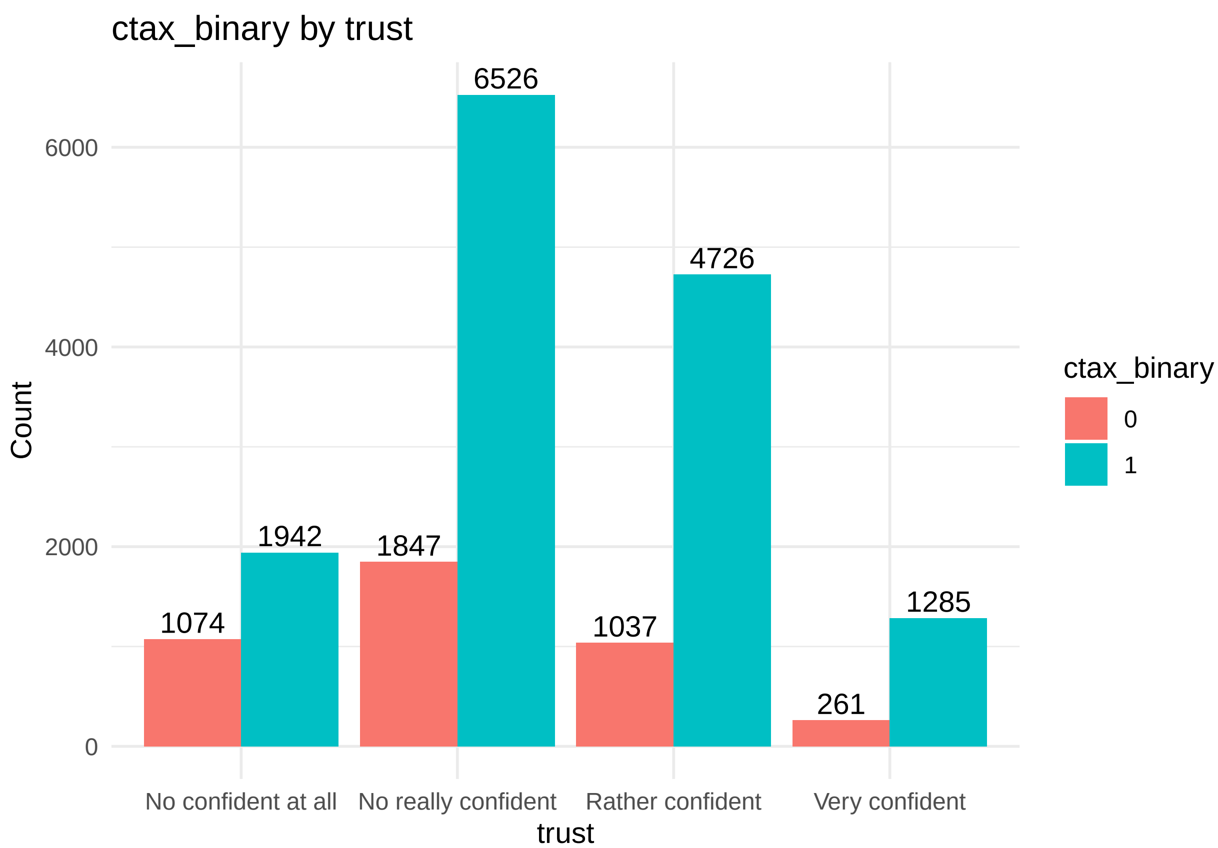
```r
# Function to create and save bar plots of ctax_binary against binary/categorical predictors
response_plots  <- function(data, response = "ctax_binary") {
  # Ensure response is a factor
  data[[response]] <- as.factor(data[[response]])

  # Loop through all variables except the response
  for (v in setdiff(names(data), response)) {

    # Skip if all NA or constant
    if (all(is.na(data[[v]])) || length(unique(data[[v]])) <= 1) next

    # Determine plot type
    if (is.numeric(data[[v]]) && length(unique(data[[v]])) > 2) {
      # Continuous: use boxplot
      p <- ggplot(data, aes_string(x = response, y = v, fill = response)) +
        geom_boxplot() +
        labs(title = paste(v, "by", response),
             x = response, y = v) +
        theme_minimal()

    } else {
      # Categorical/binary: use bar plot
      p <- ggplot(data, aes_string(x = v, fill = response)) +
        geom_bar(position = "dodge") +
```

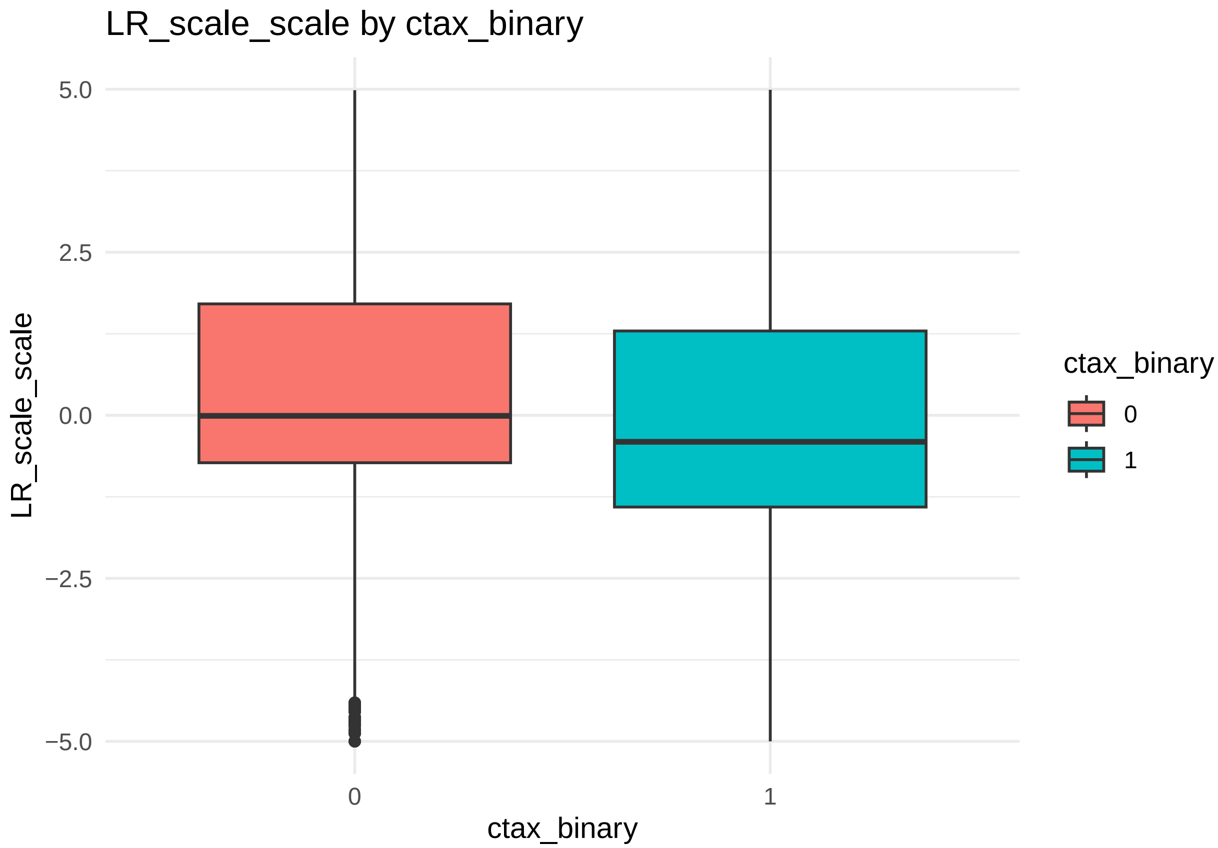
```r
        geom_text(stat = "count", aes(label = ..count..),
                  position = position_dodge(width = 0.9), vjust = -0.3) +
        labs(title = paste("ctax_binary by", v),
             x = v, y = "Count", fill = response) +
        theme_minimal()
    }

    print(p)
  }
}
```

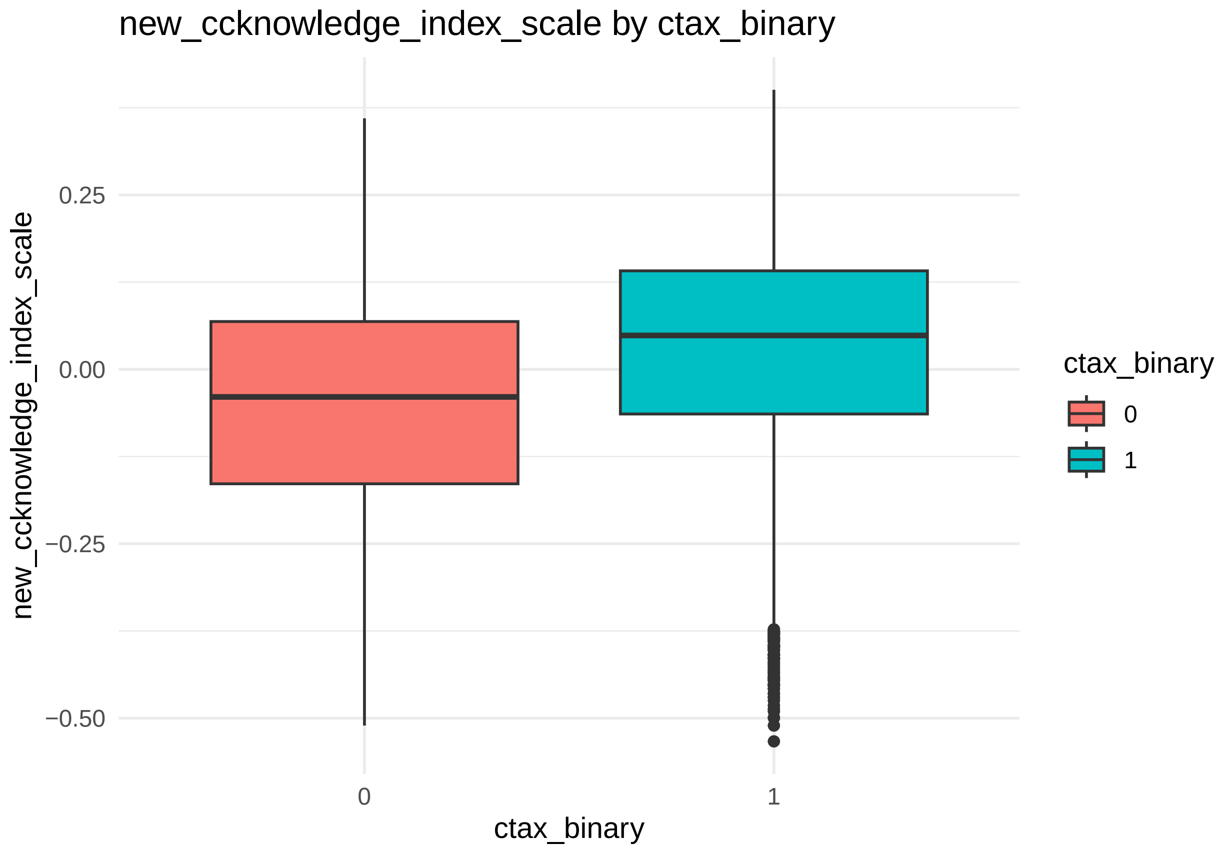
```r
response_plots(df_clean,response="ctax_binary")
```

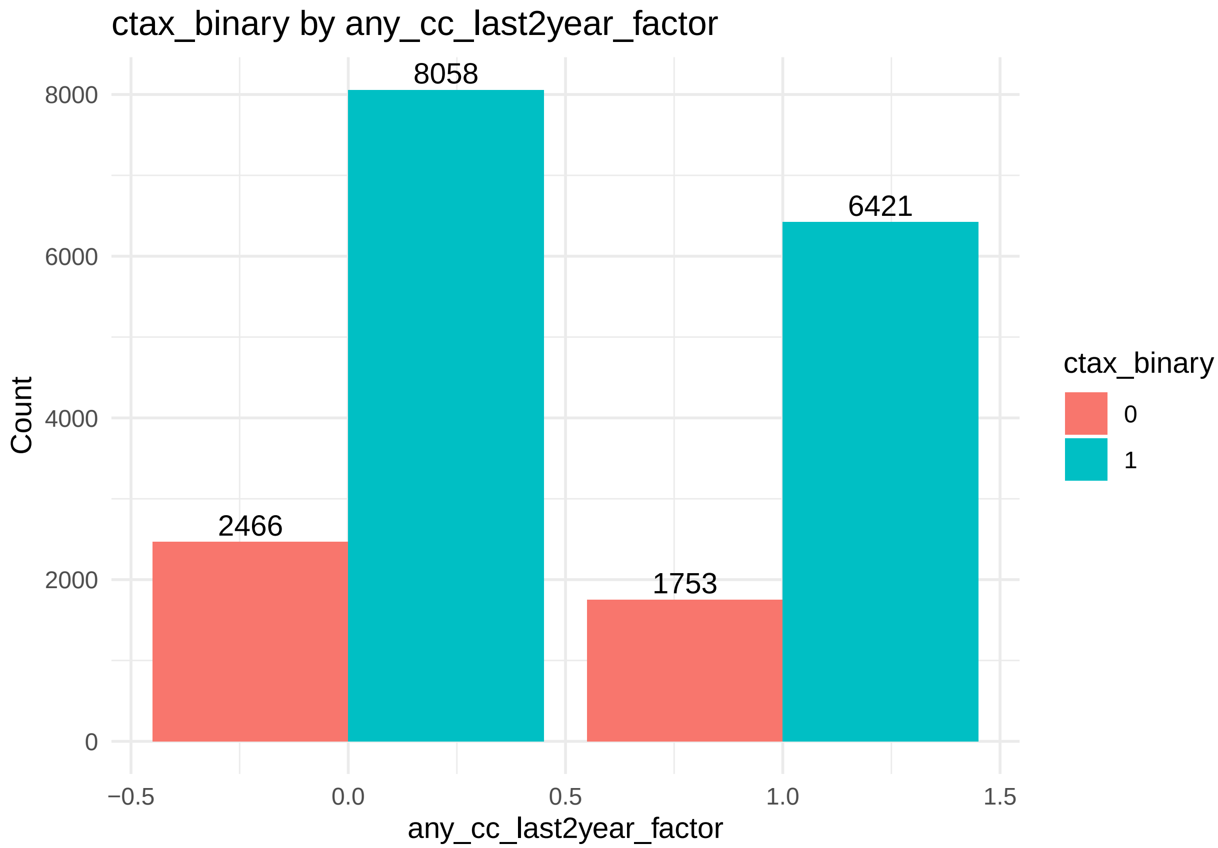


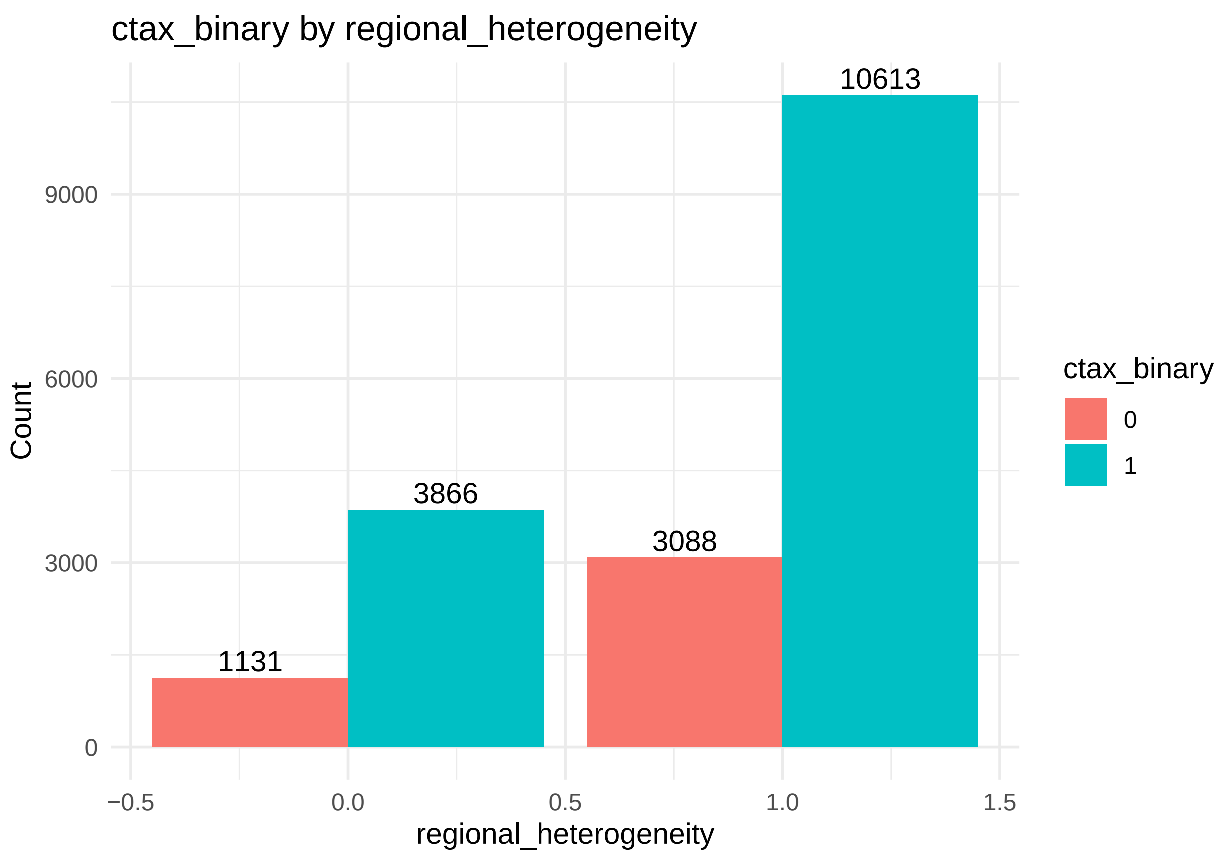

age_scale by ctax_binary
40
20
0
−20
age_scale
0
1
ctax_binary
ctax_binary
0
1

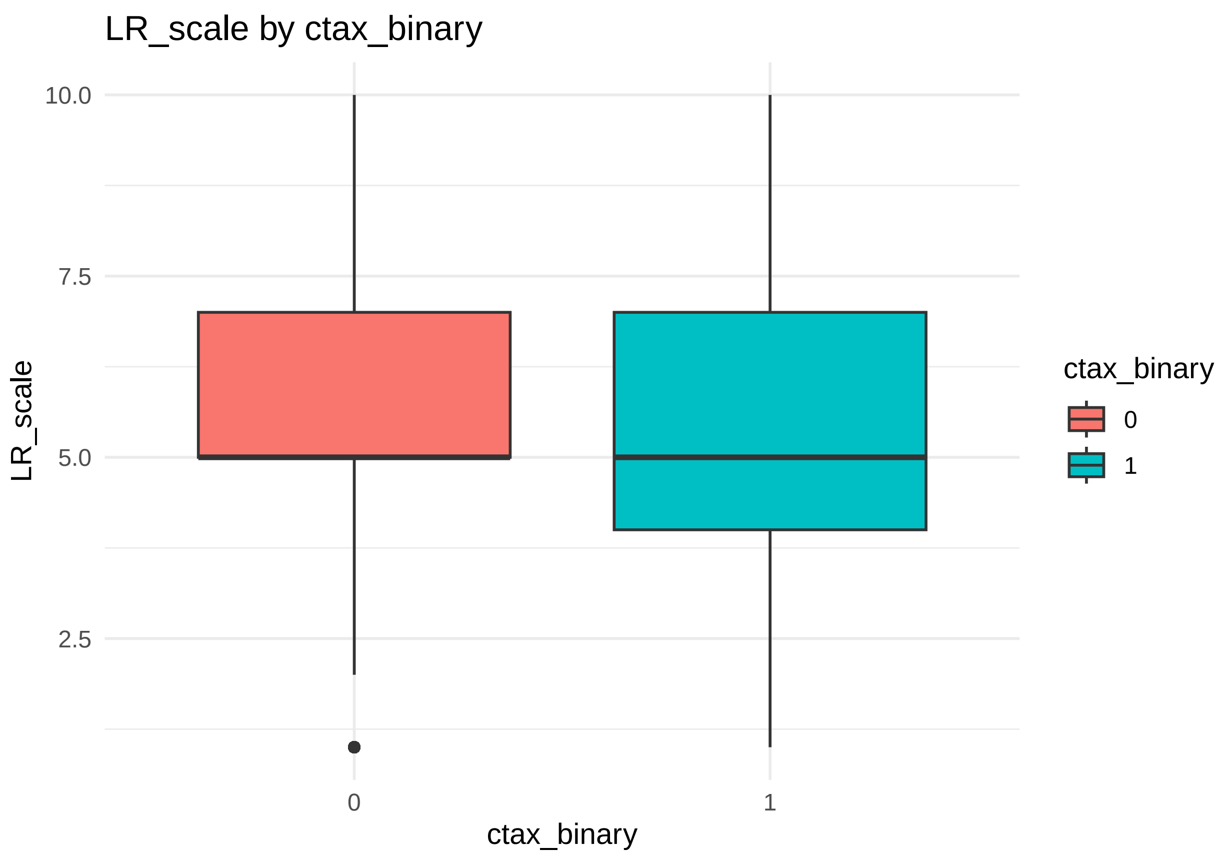

ctax_binary by trust
6000
4000
2000
0
Count
1074
1942
1847
6526
1037
4726
261
1285
No confident at all
No really confident
Rather confident
Very confident
trust
ctax_binary
0
1

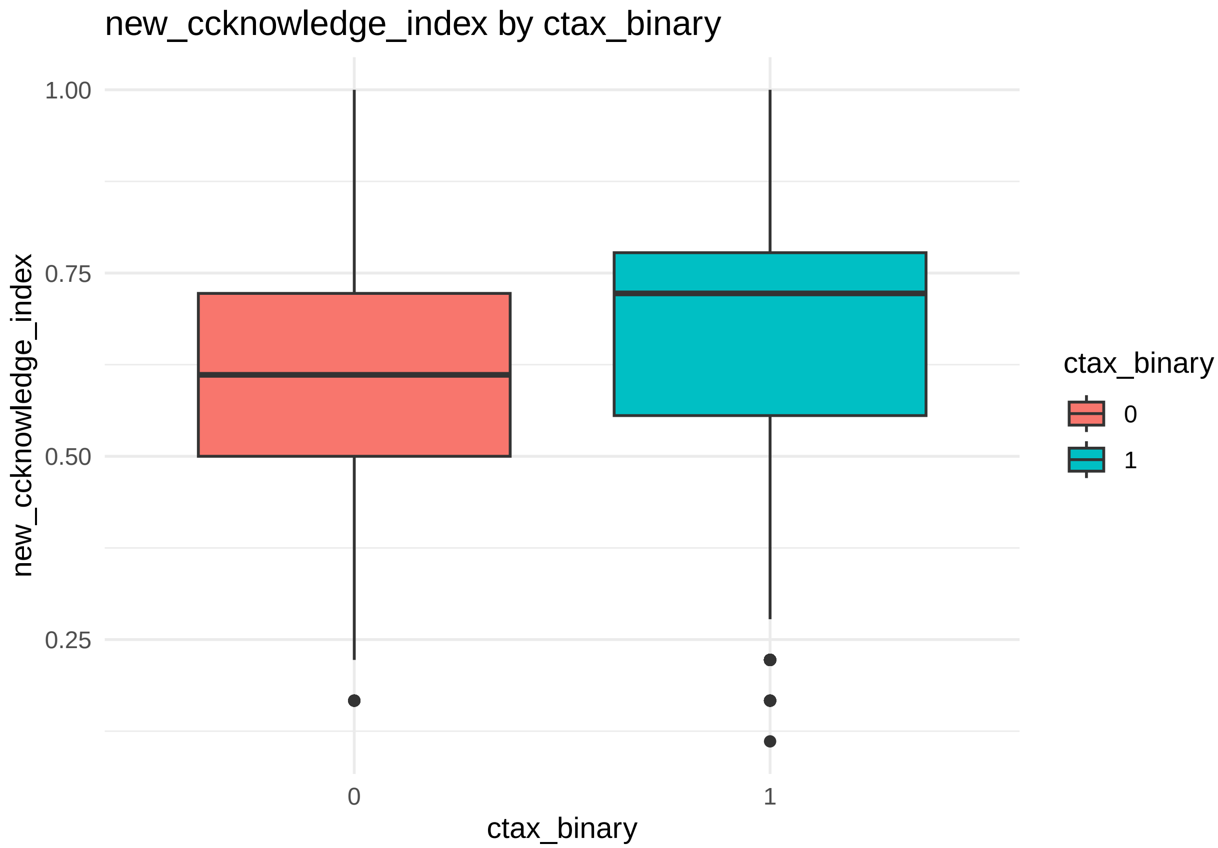

LR_scale_scale by ctax_binary
5.0
2.5
0.0
−2.5
−5.0
LR_scale_scale
0
1
ctax_binary
ctax_binary
0
1

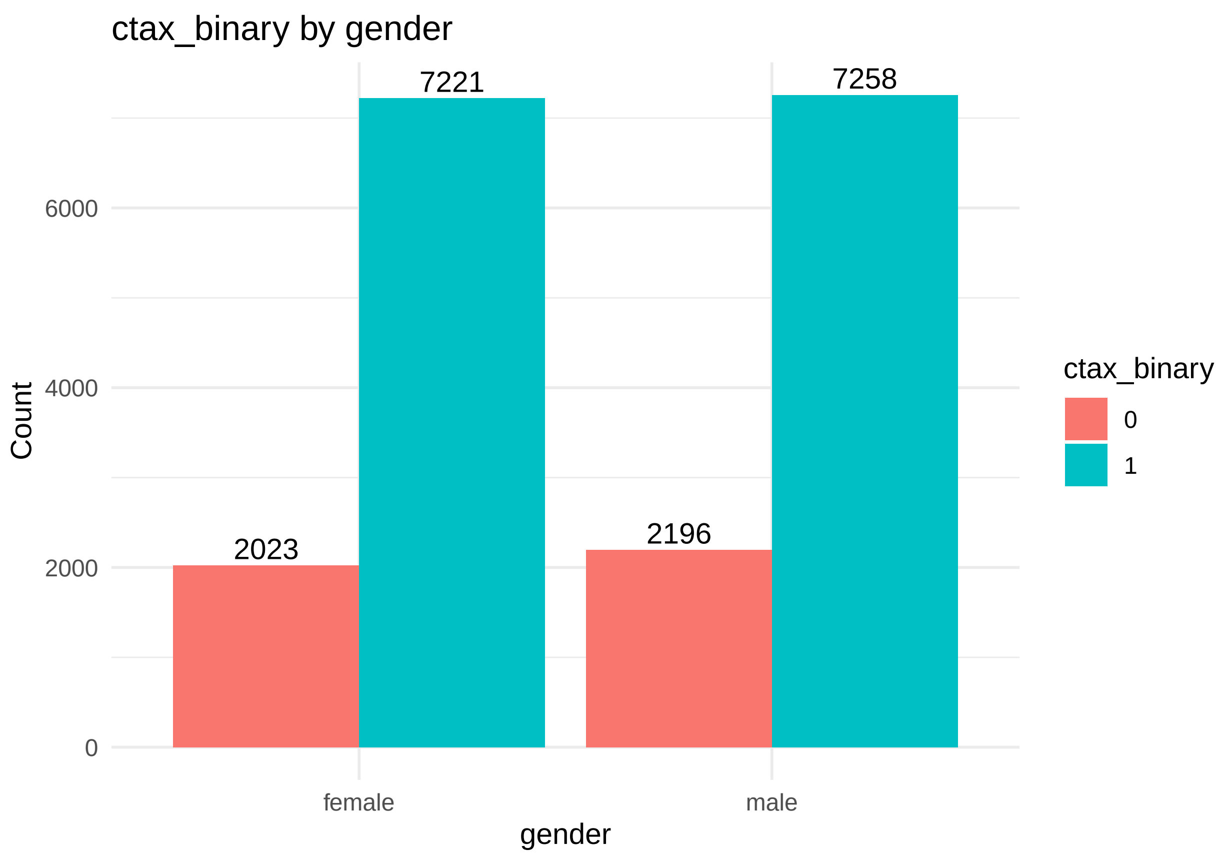

new_ccknowledge_index_scale by ctax_binary
new_ccknowledge_index_scale
0.25
0.00
−0.25
−0.50
0
1
ctax_binary
ctax_binary
0
1


ctax_binary by any_cc_last2year_factor
8058
6421
2466
1753
8000
6000
4000
2000
0
Count
−0.5
0.0
0.5
1.0
1.5
any_cc_last2year_factor
ctax_binary
0
1

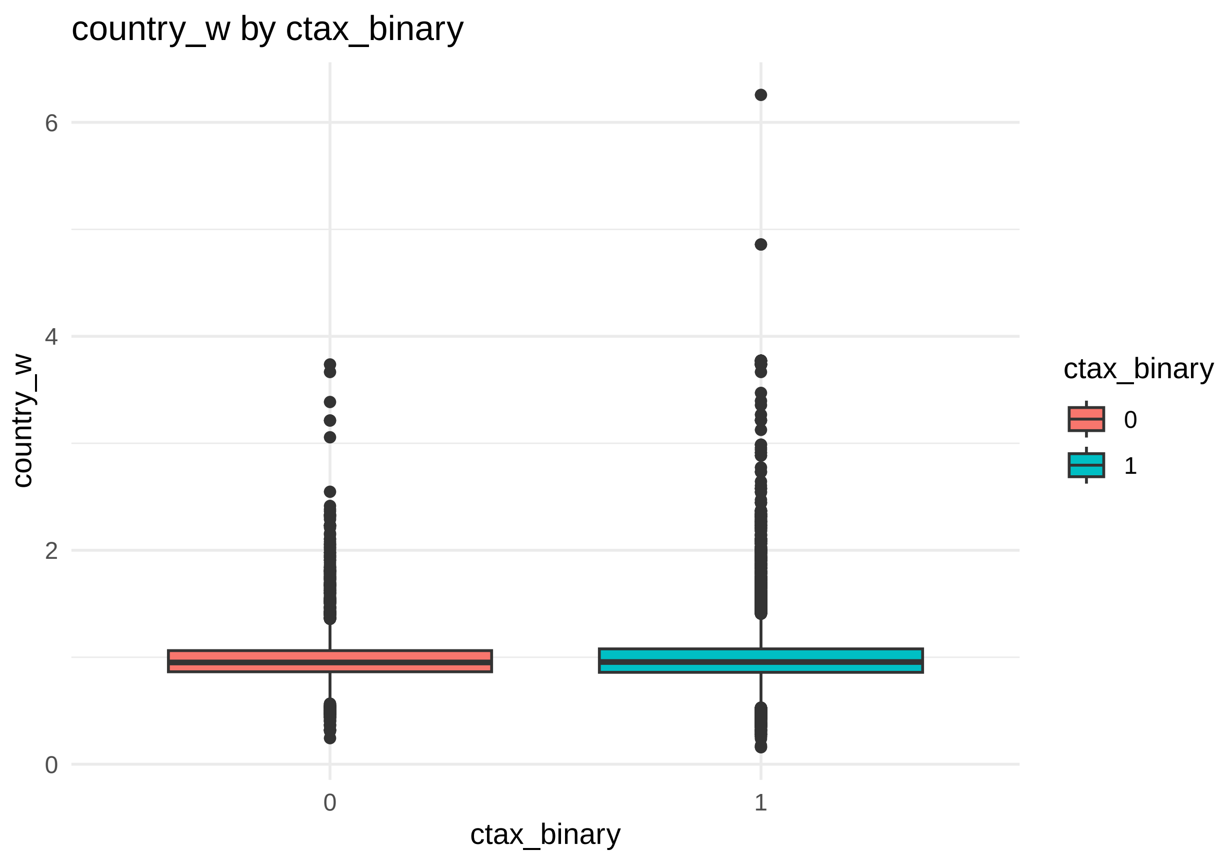

ctax_binary by regional_heterogeneity
10613
3866
3088
1131
9000
6000
3000
0
Count
−0.5
0.0
0.5
1.0
1.5
regional_heterogeneity
ctax_binary
0
1

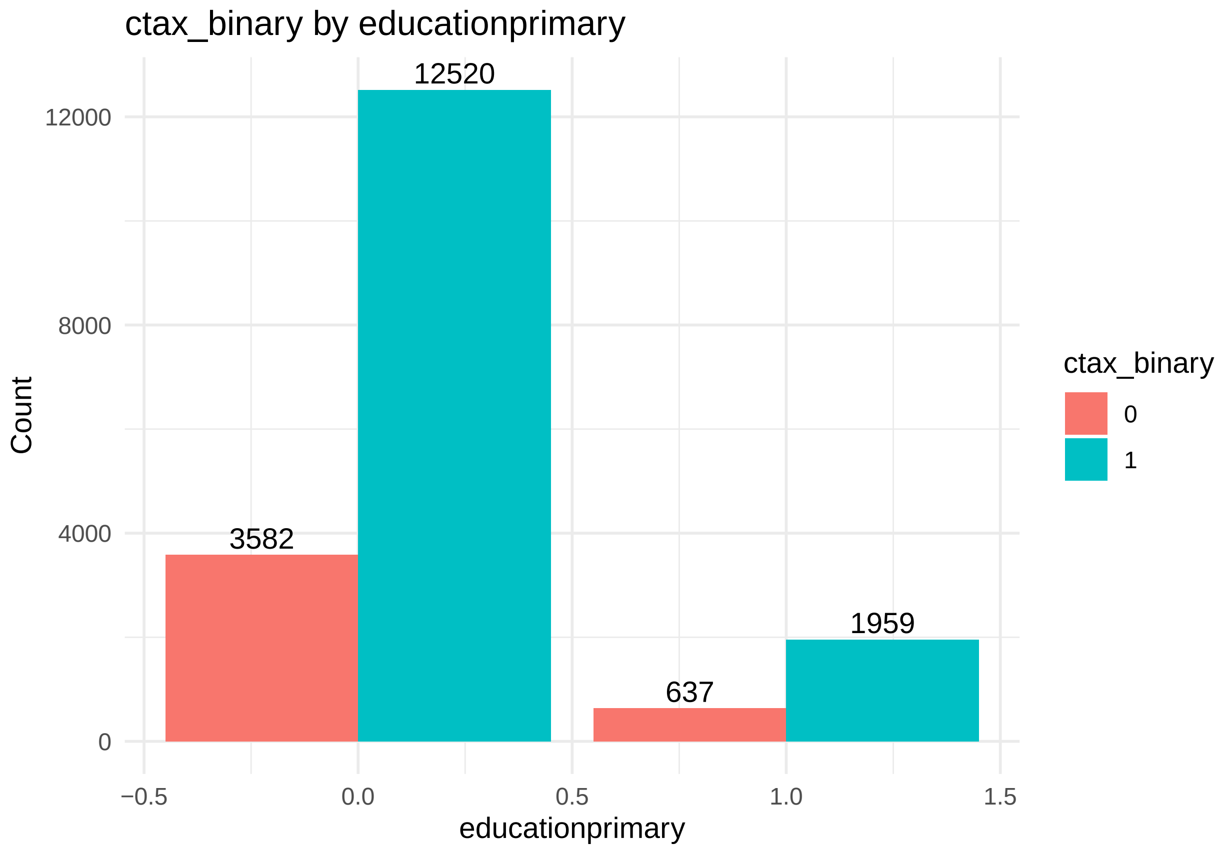

LR_scale by ctax_binary
10.0
7.5
5.0
2.5
LR_scale
0
1
ctax_binary
ctax_binary
0
1

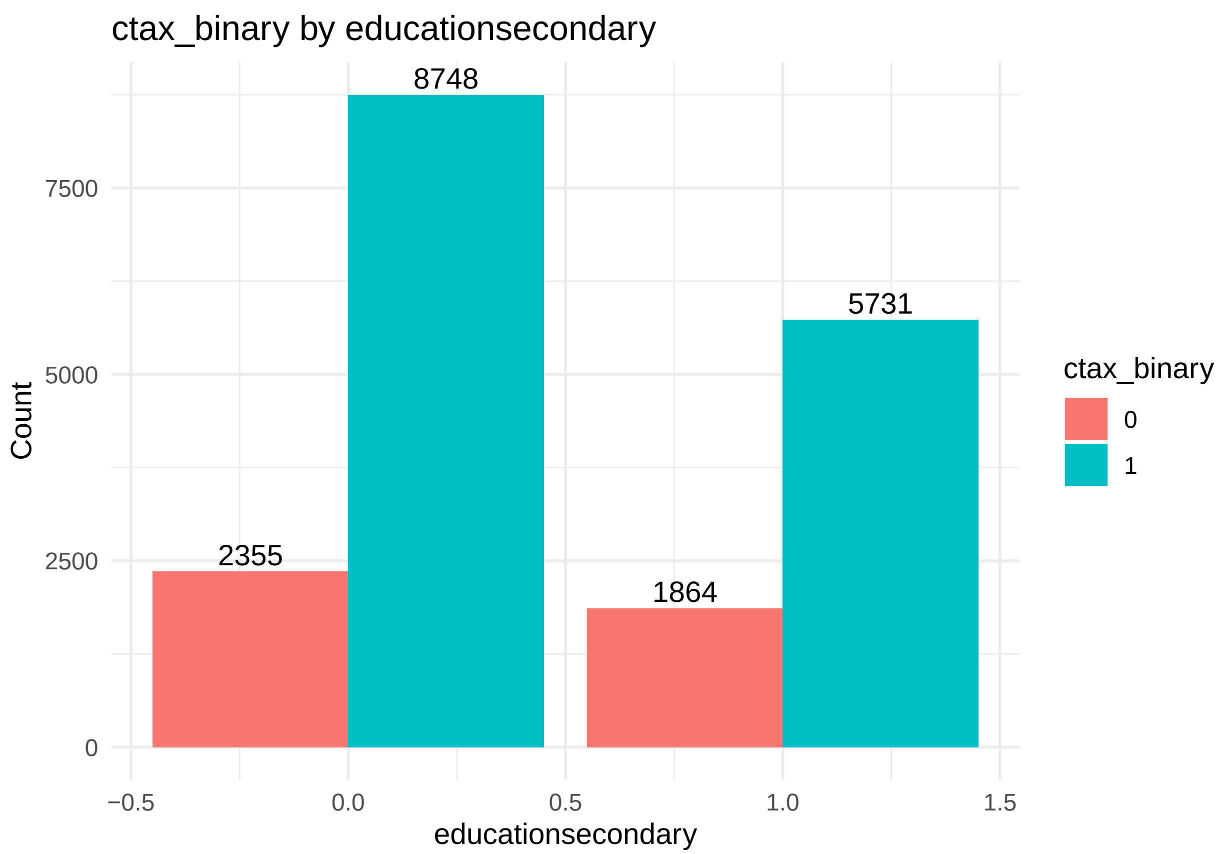

new_ccknowledge_index by ctax_binary
1.00
0.75
0.50
0.25
new_ccknowledge_index
0
1
ctax_binary
ctax_binary
0
1

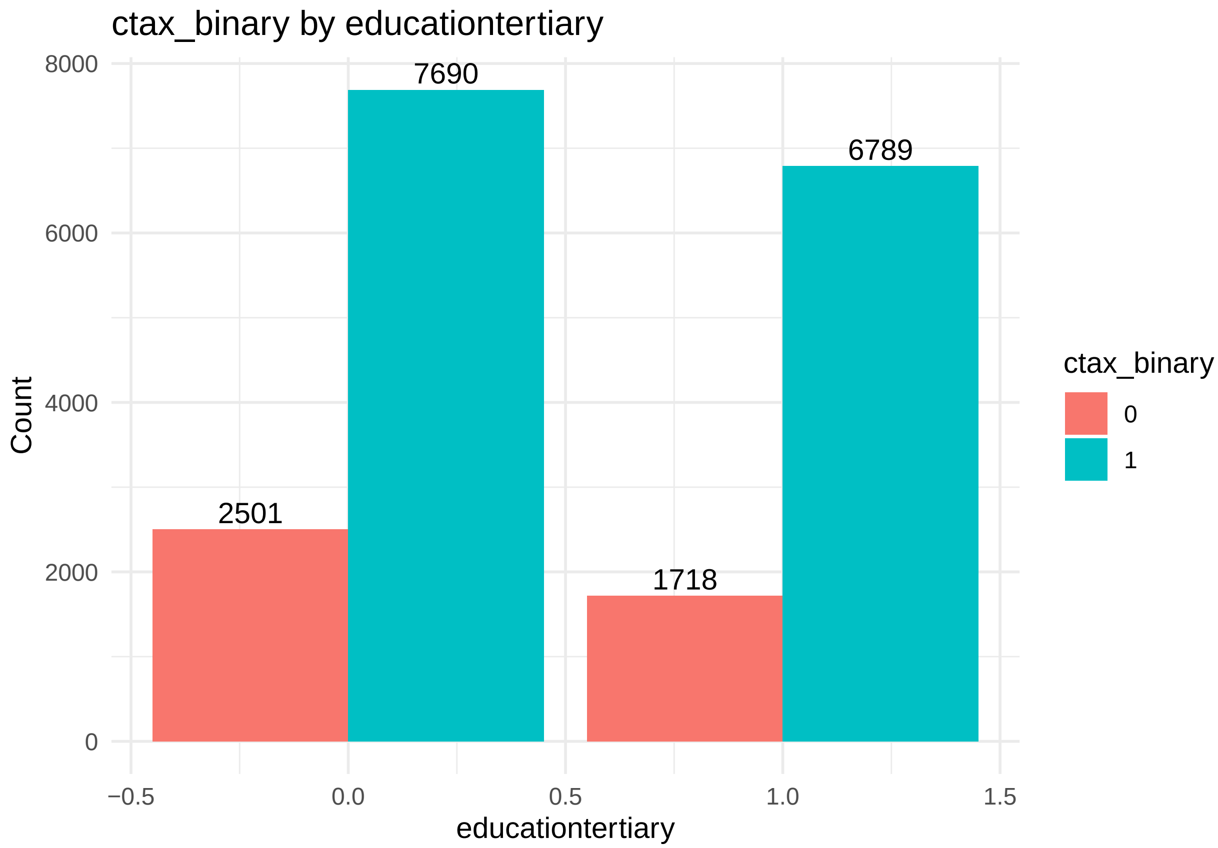


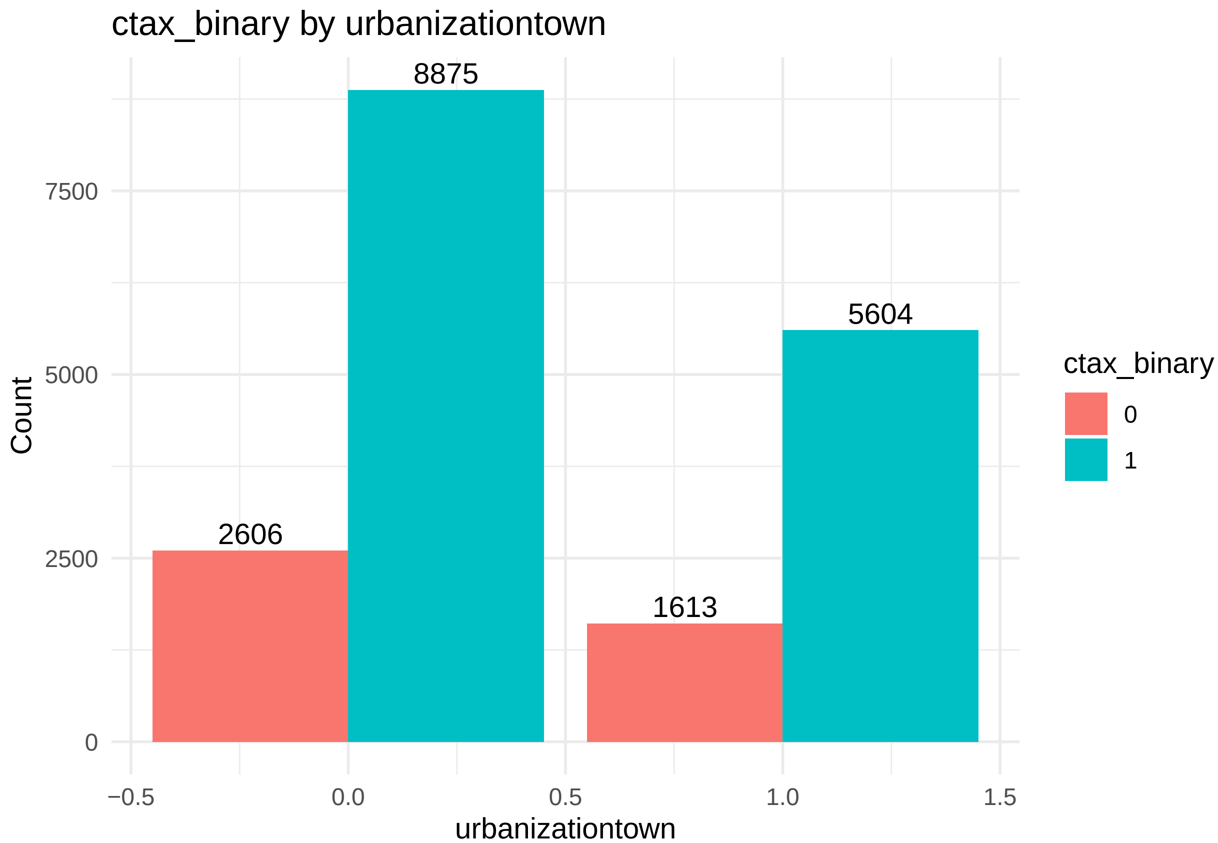
ctax_binary by gender
7221
7258
2023
2196
6000
4000
2000
0
Count
female
male
gender
ctax_binary
0
1

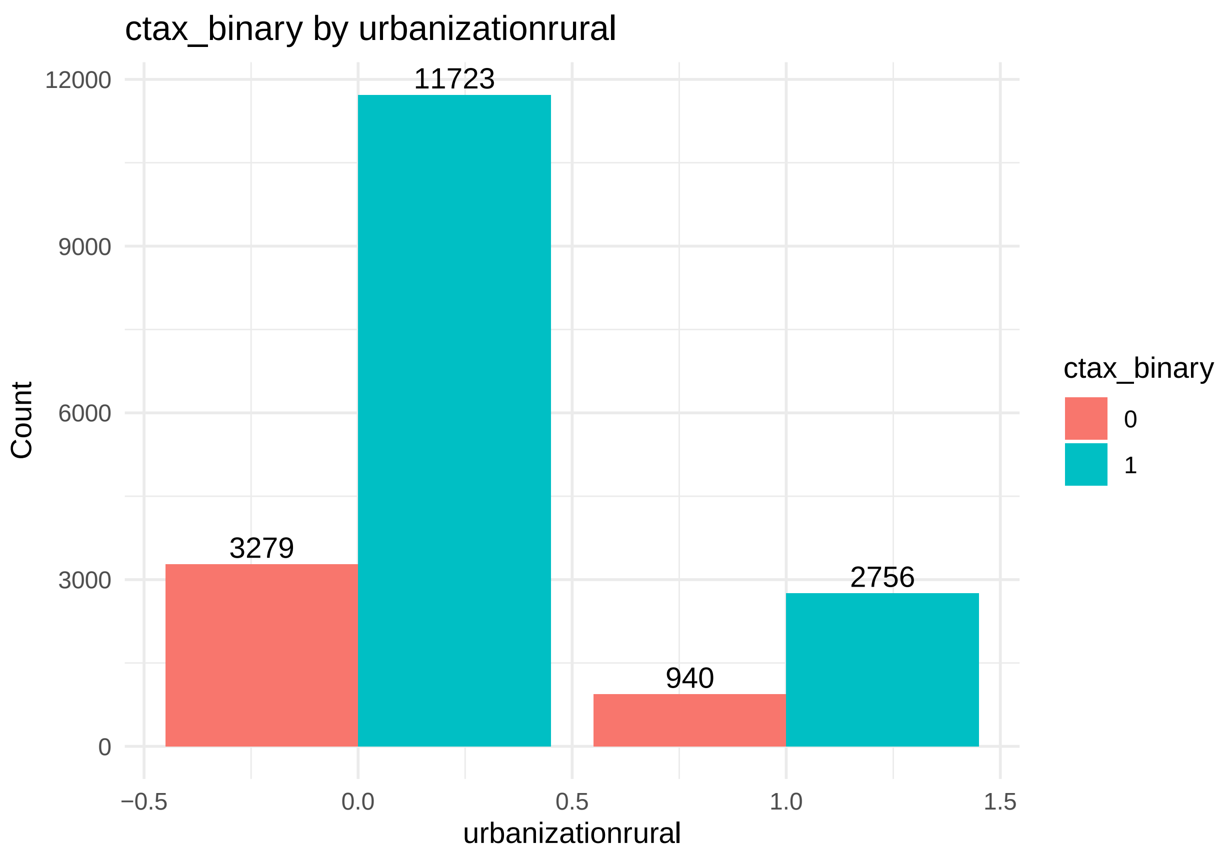

country_w by ctax_binary
country_w
6
4
2
0
0
1
ctax_binary
ctax_binary
0
1

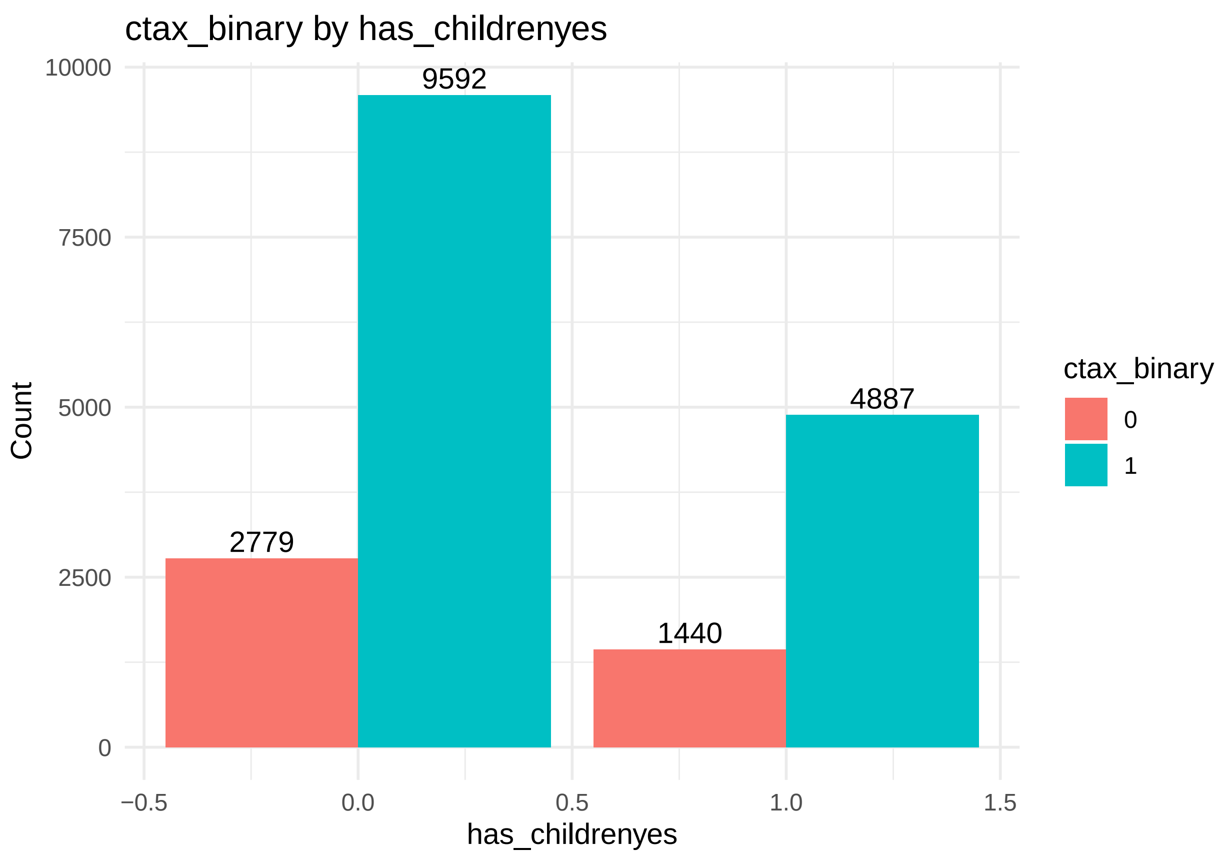

ctax_binary by educationprimary
12520
3582
637
1959
12000
8000
4000
0
Count
−0.5
0.0
0.5
1.0
1.5
educationprimary
ctax_binary
0
1

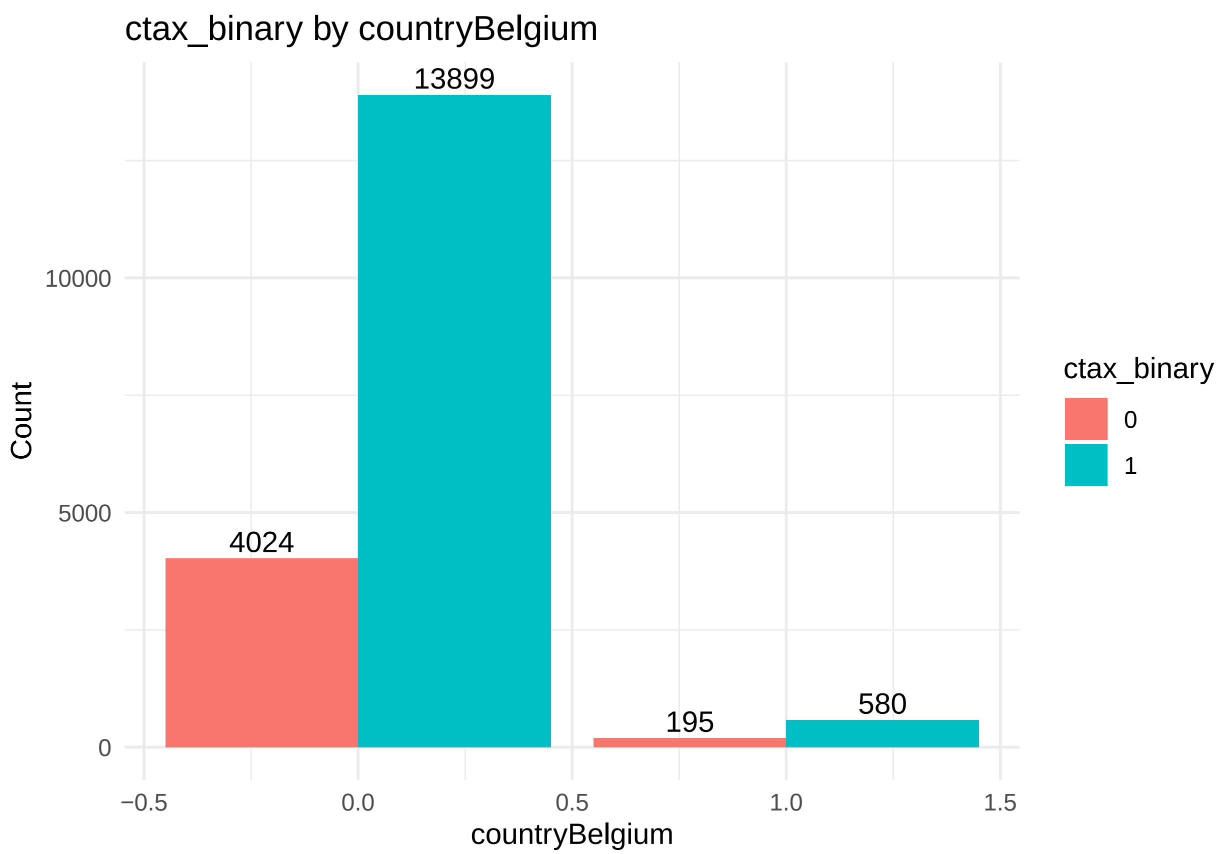

ctax_binary by educationsecondary
8748
2355
5731
1864
Count
7500
5000
2500
0
−0.5
0.0
0.5
1.0
1.5
educationsecondary
ctax_binary
0
1

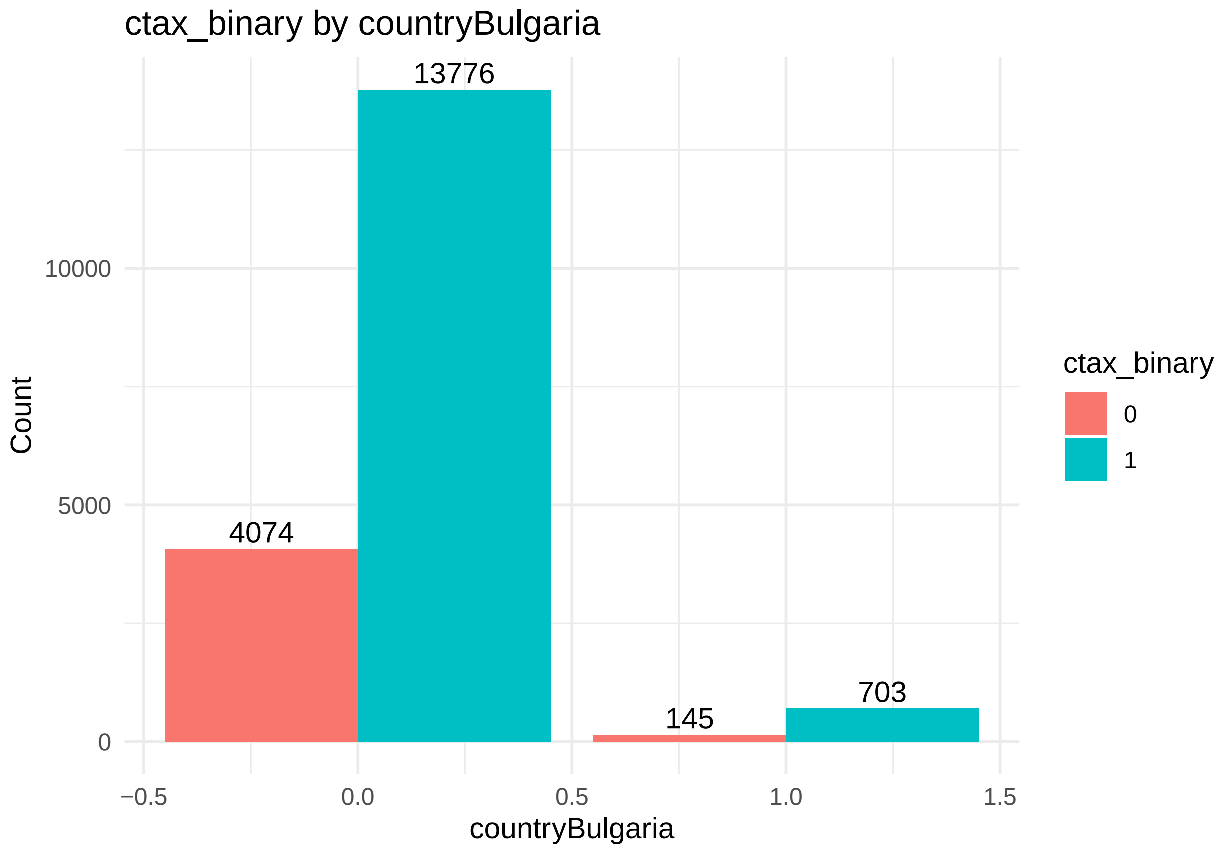

ctax_binary by educationtertiary
8000
6000
4000
2000
0
Count
7690
6789
2501
1718
−0.5
0.0
0.5
1.0
1.5
educationtertiary
ctax_binary
0
1


ctax_binary by urbanizationtown
8875
7500
5604
5000
Count
ctax_binary
0
1
2606
2500
1613
0
−0.5
0.0
0.5
1.0
1.5
urbanizationtown

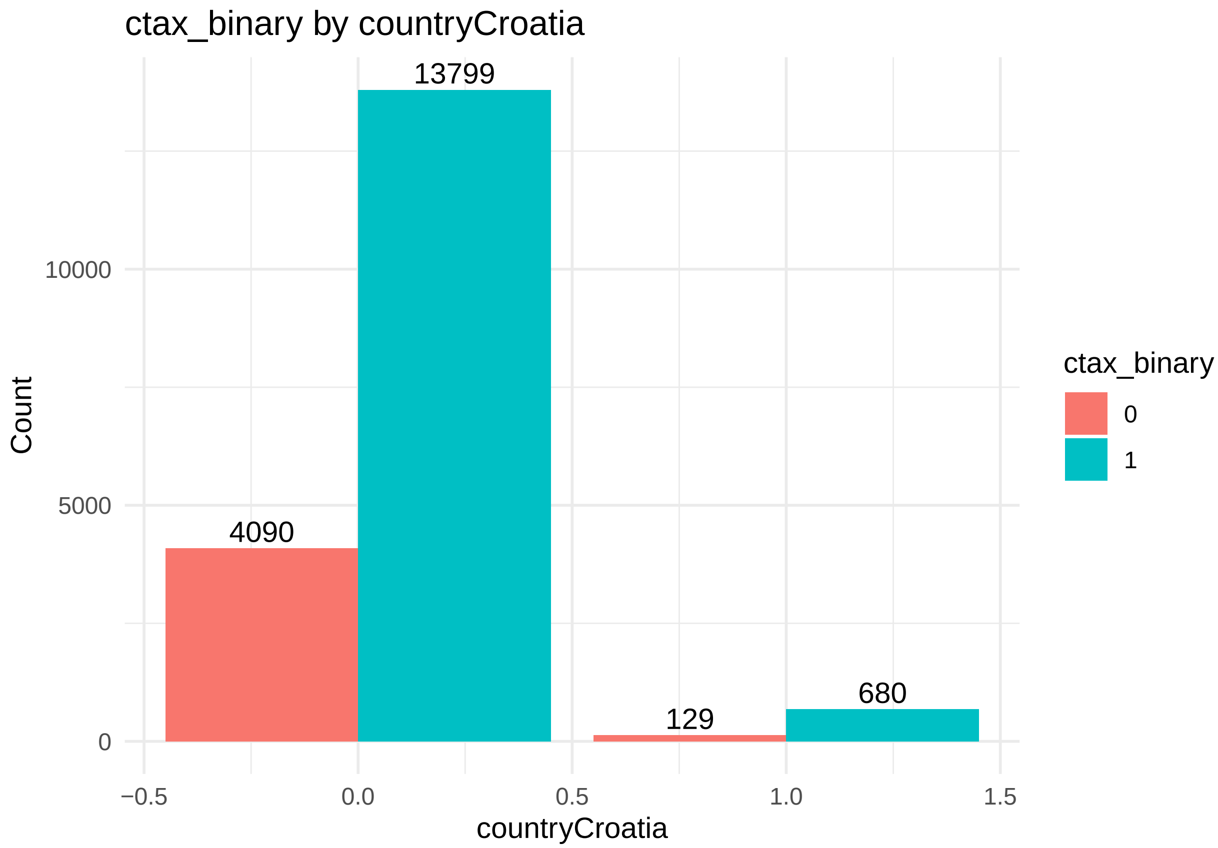

ctax_binary by urbanizationrural
Count
12000
9000
6000
3000
0
11723
3279
2756
940
−0.5
0.0
0.5
1.0
1.5
urbanizationrural
ctax_binary
0
1

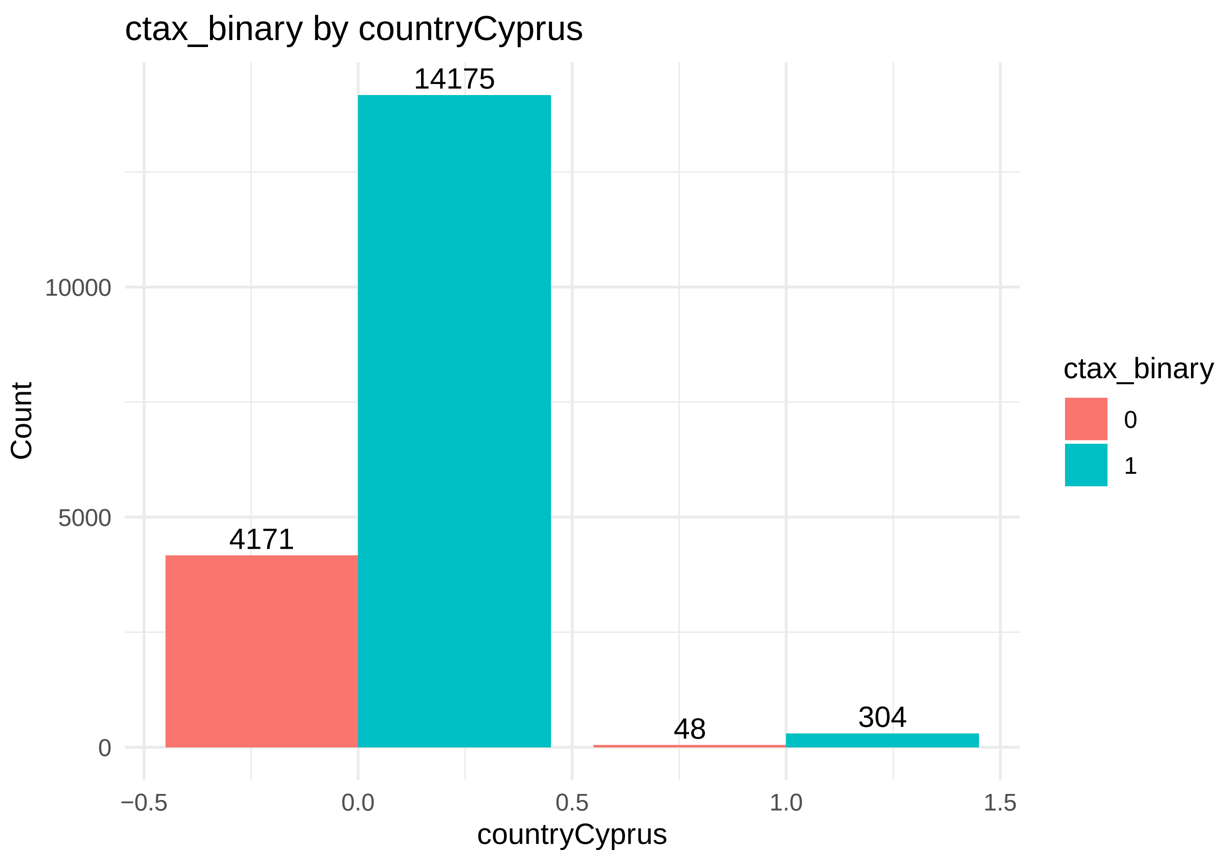

ctax_binary by has_childrenyes
Count
10000
7500
5000
2500
0
9592
2779
4887
1440
−0.5
0.0
0.5
1.0
1.5
has_childrenyes
ctax_binary
0
1

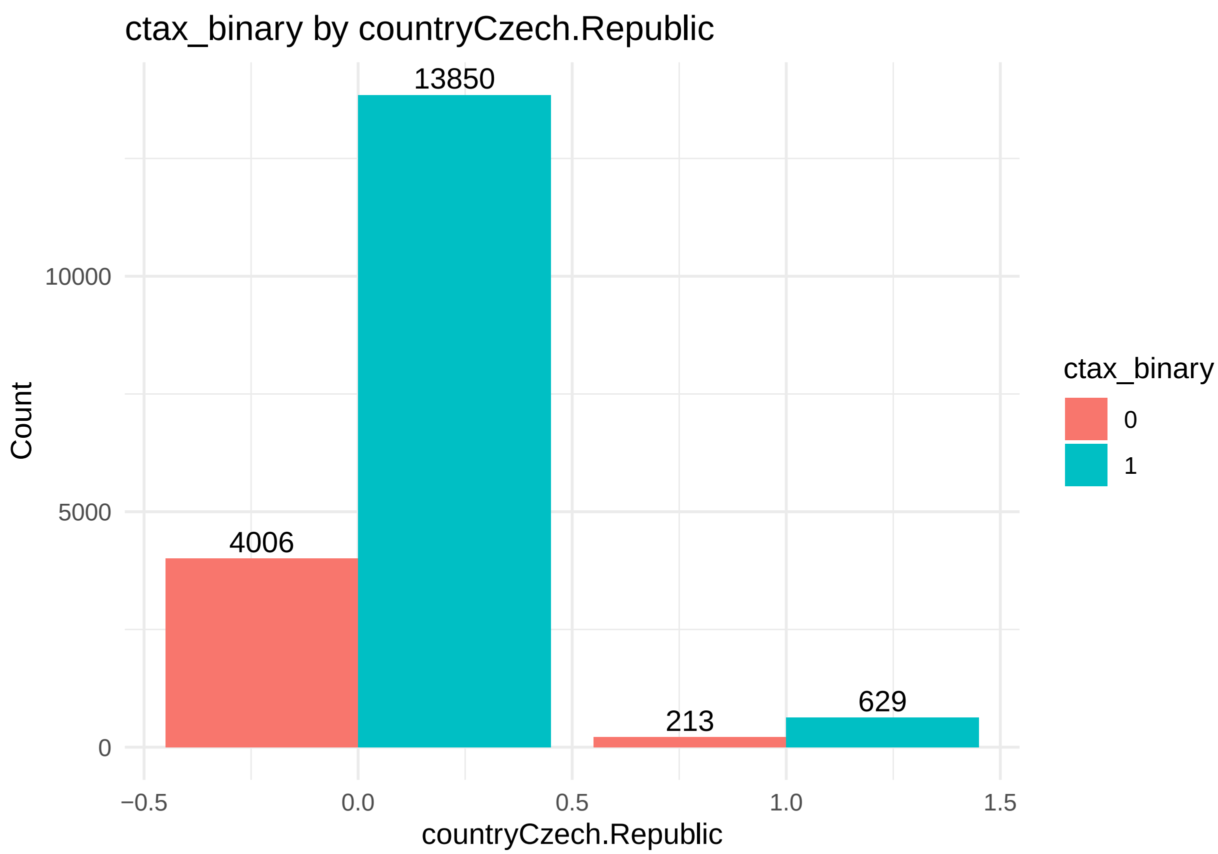

ctax_binary by countryBelgium
13899
4024
195
580
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryBelgium
ctax_binary
0
1

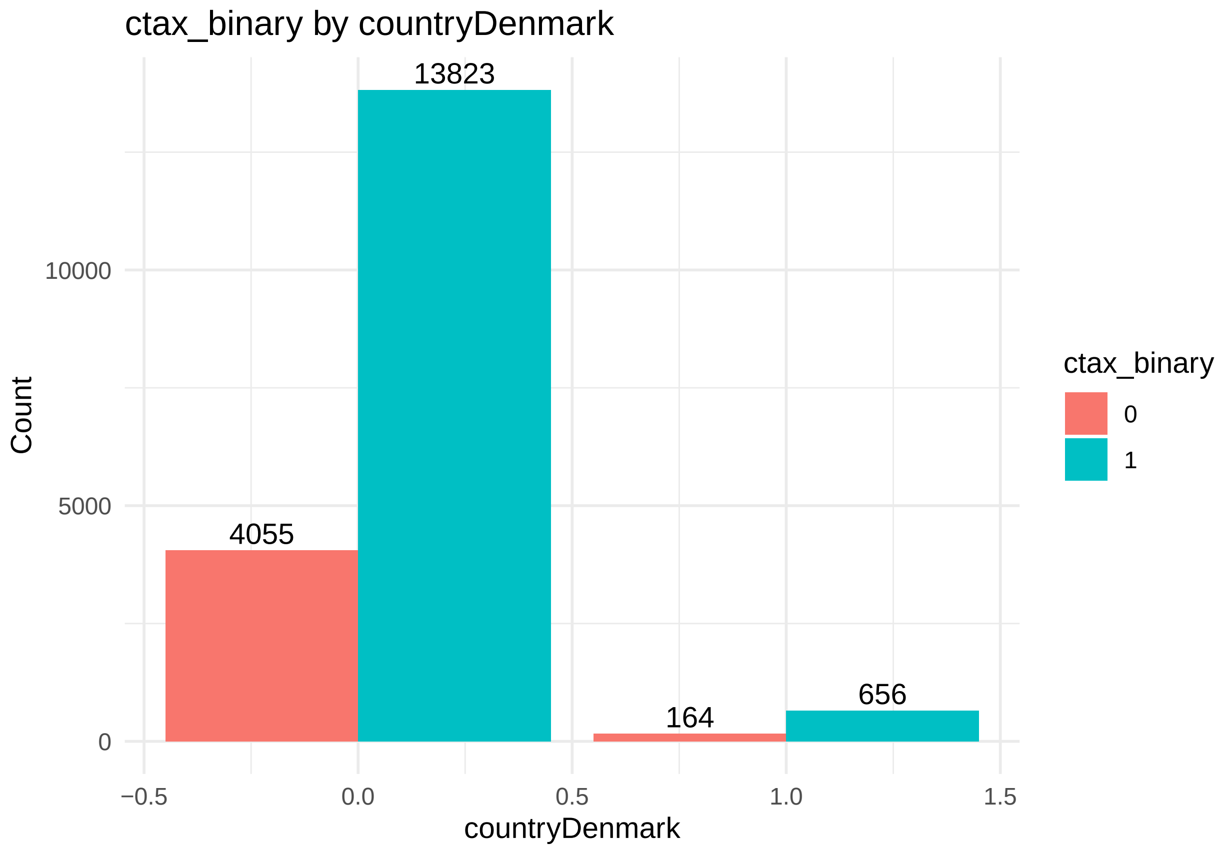


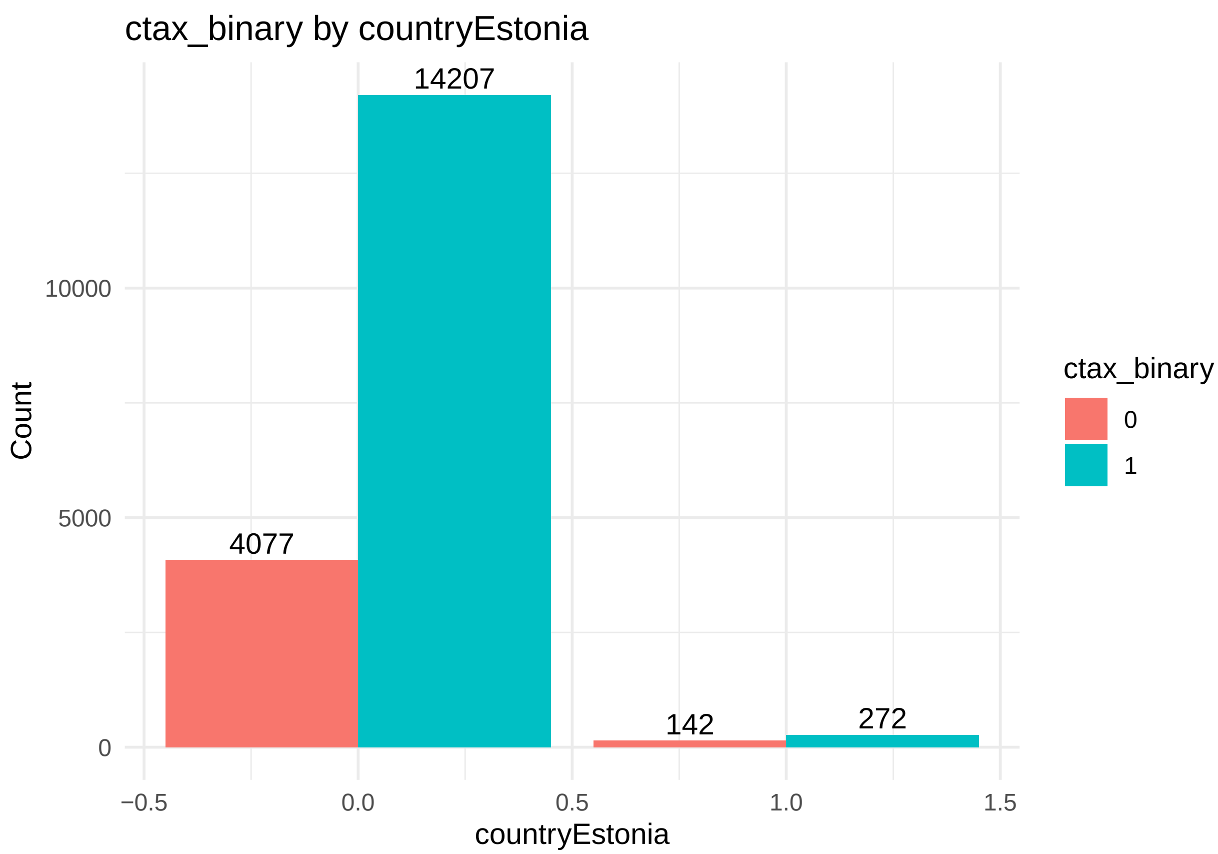
ctax_binary by countryBulgaria
13776
4074
145
703
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryBulgaria
ctax_binary
0
1

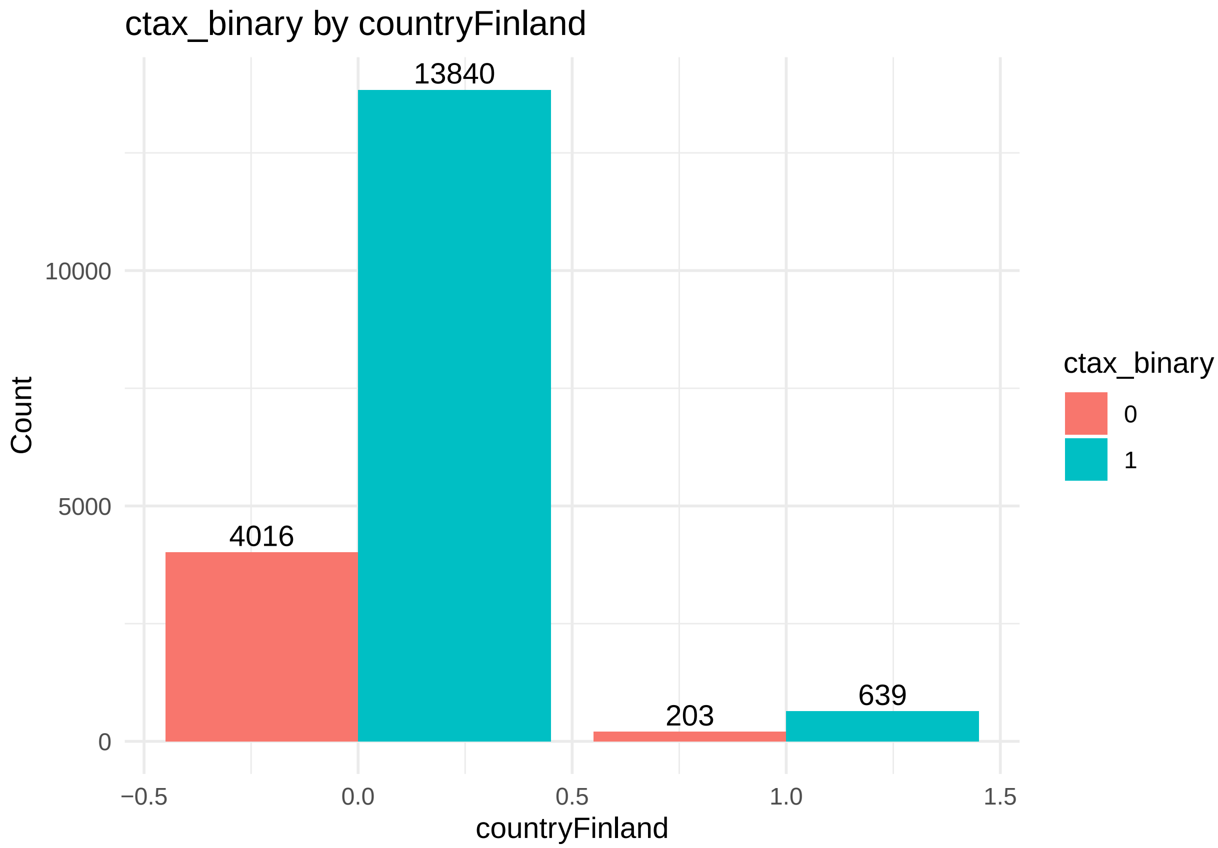

ctax_binary by countryCroatia
13799
4090
129
680
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryCroatia
ctax_binary
0
1


ctax_binary by countryCyprus
14175
4171
48
304
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryCyprus
ctax_binary
0
1

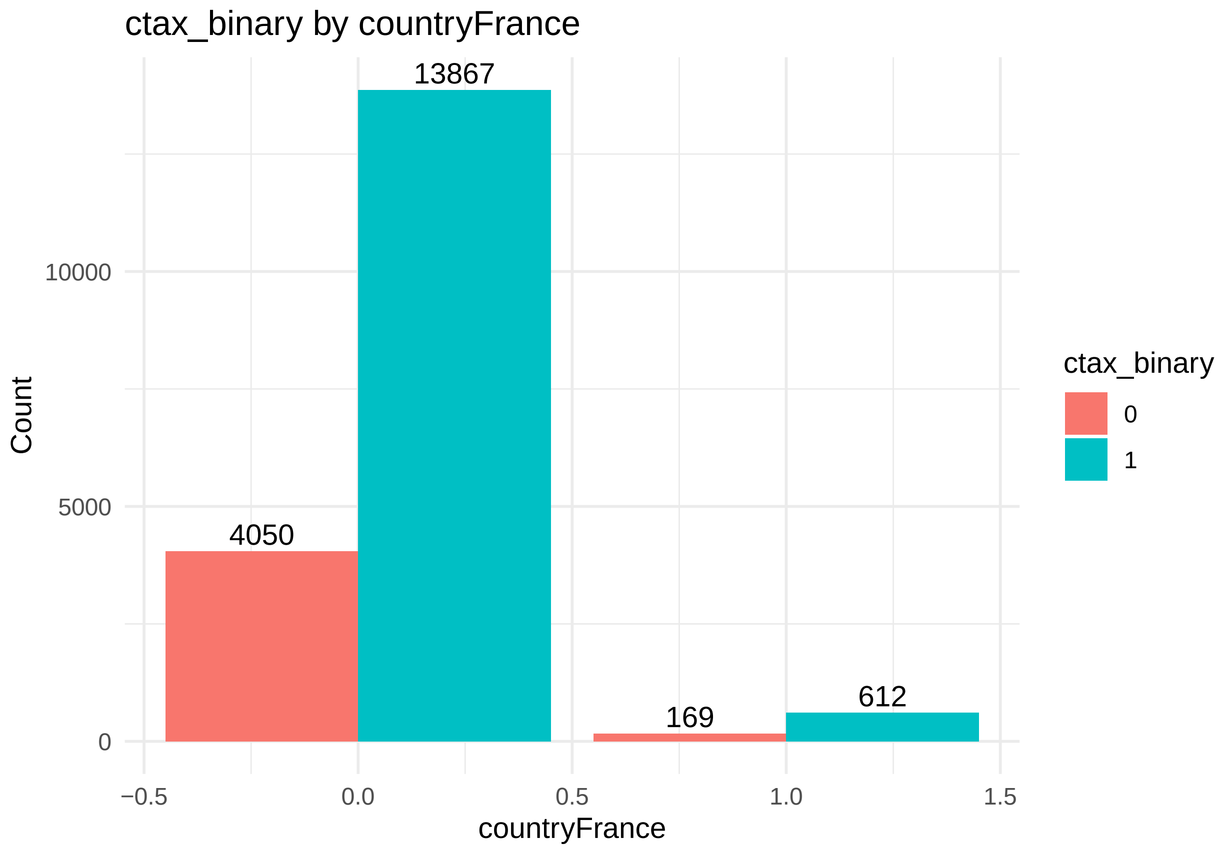

ctax_binary by countryCzech.Republic
13850
4006
213
629
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryCzech.Republic
ctax_binary
0
1

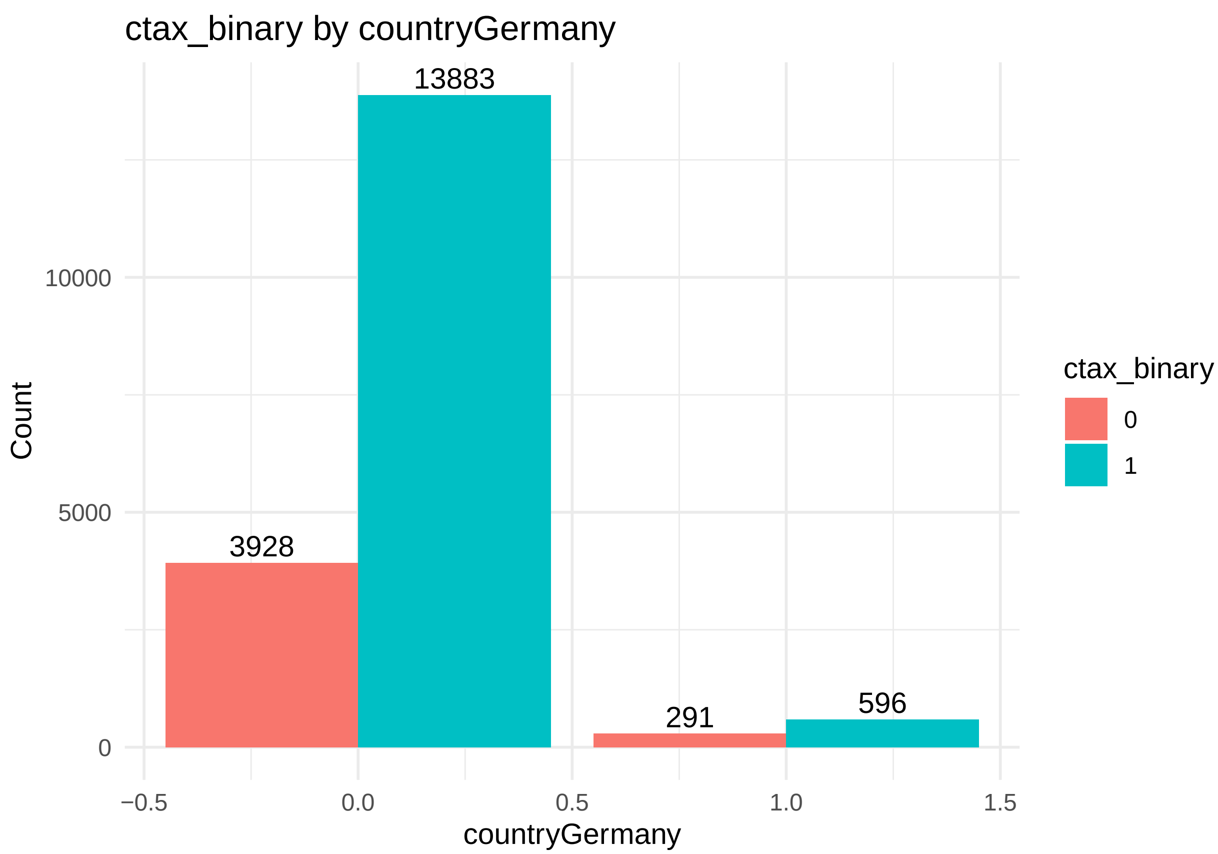

ctax_binary by countryDenmark
13823
4055
164
656
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryDenmark
ctax_binary
0
1


ctax_binary by countryEstonia
14207
4077
142
272
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryEstonia
ctax_binary
0
1

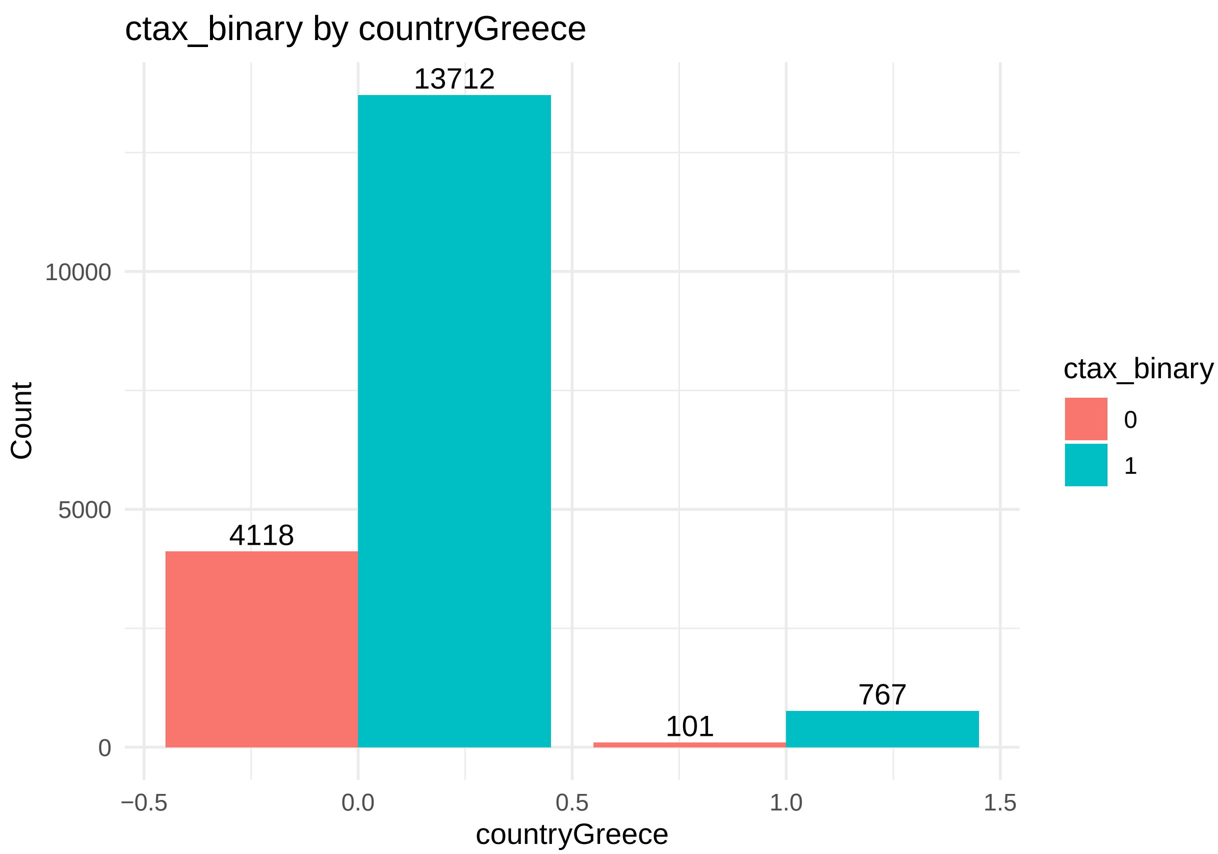


ctax_binary by countryFinland
13840
4016
203
639
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryFinland
ctax_binary
0
1

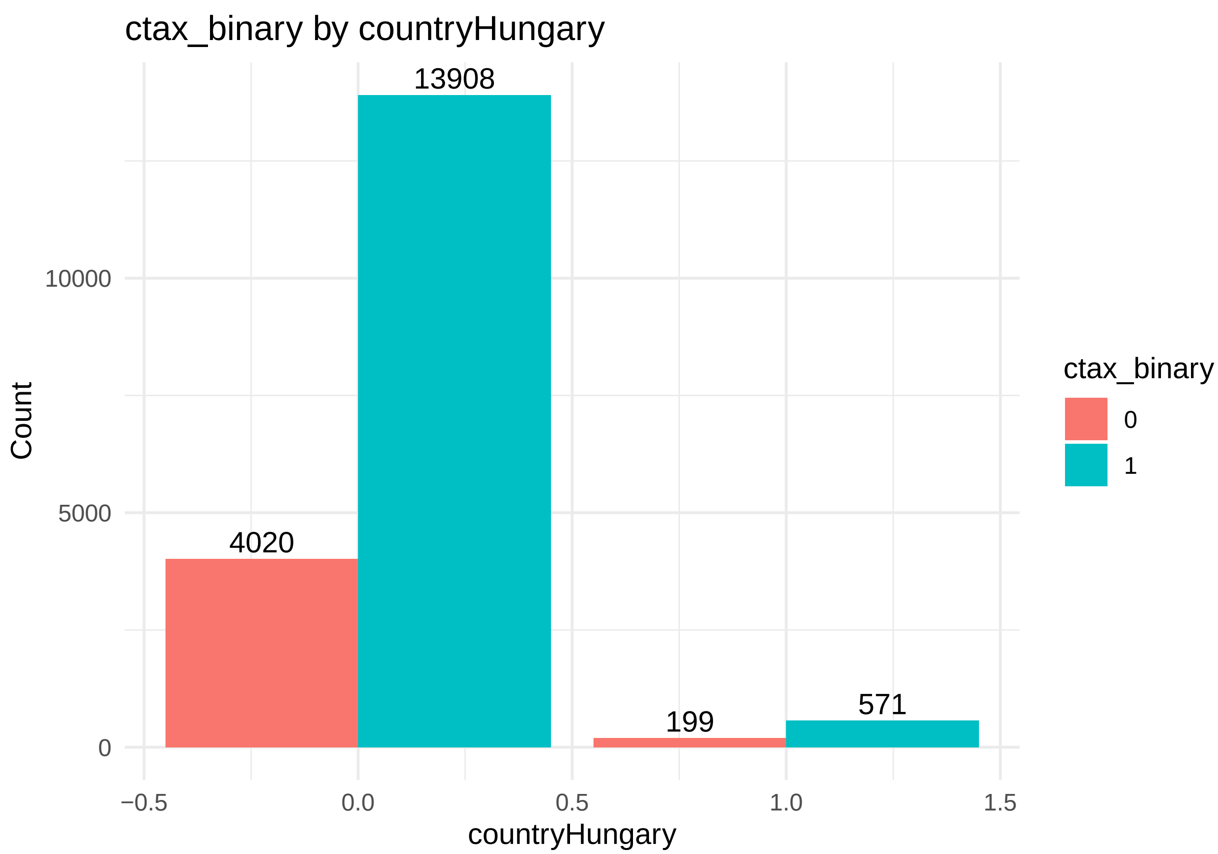

ctax_binary by countryFrance
13867
4050
169
612
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryFrance
ctax_binary
0
1

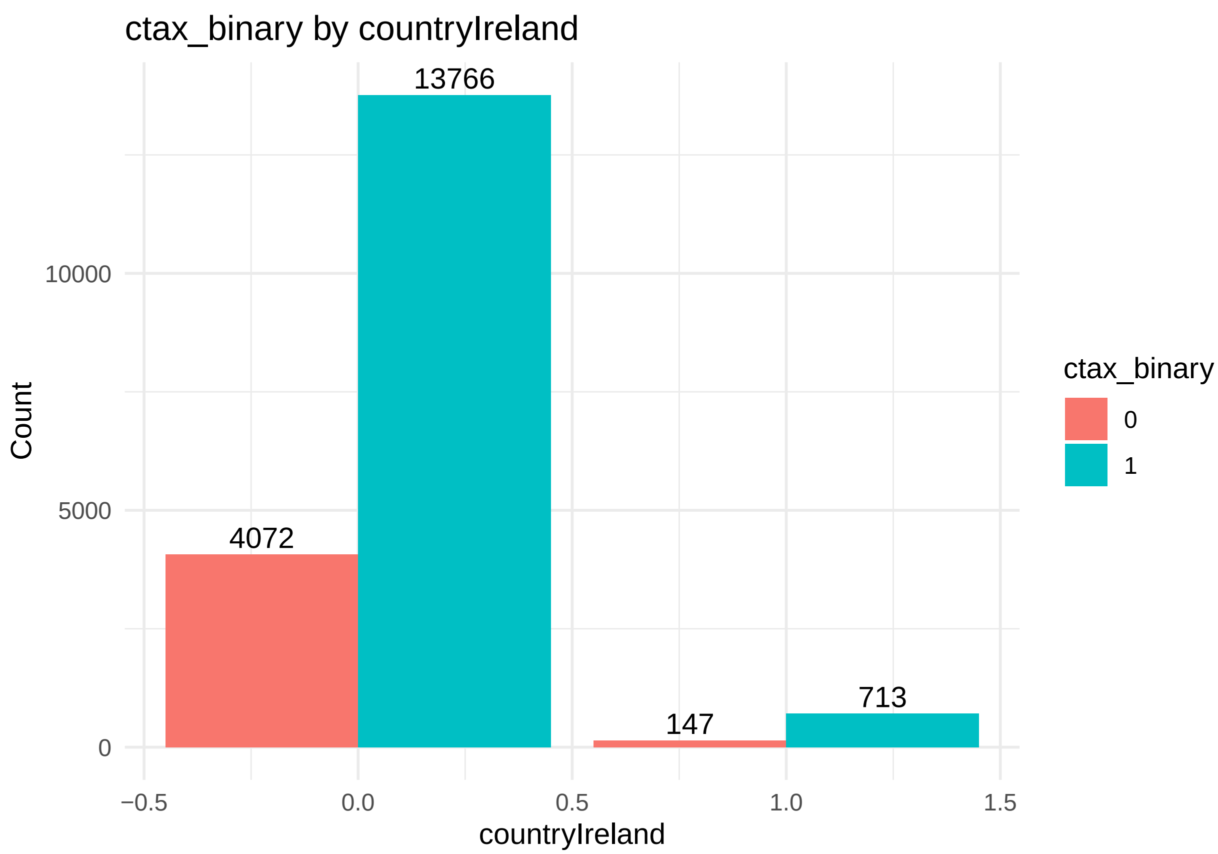


ctax_binary by countryGermany
13883
3928
291
596
Count
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryGermany
ctax_binary
0
1

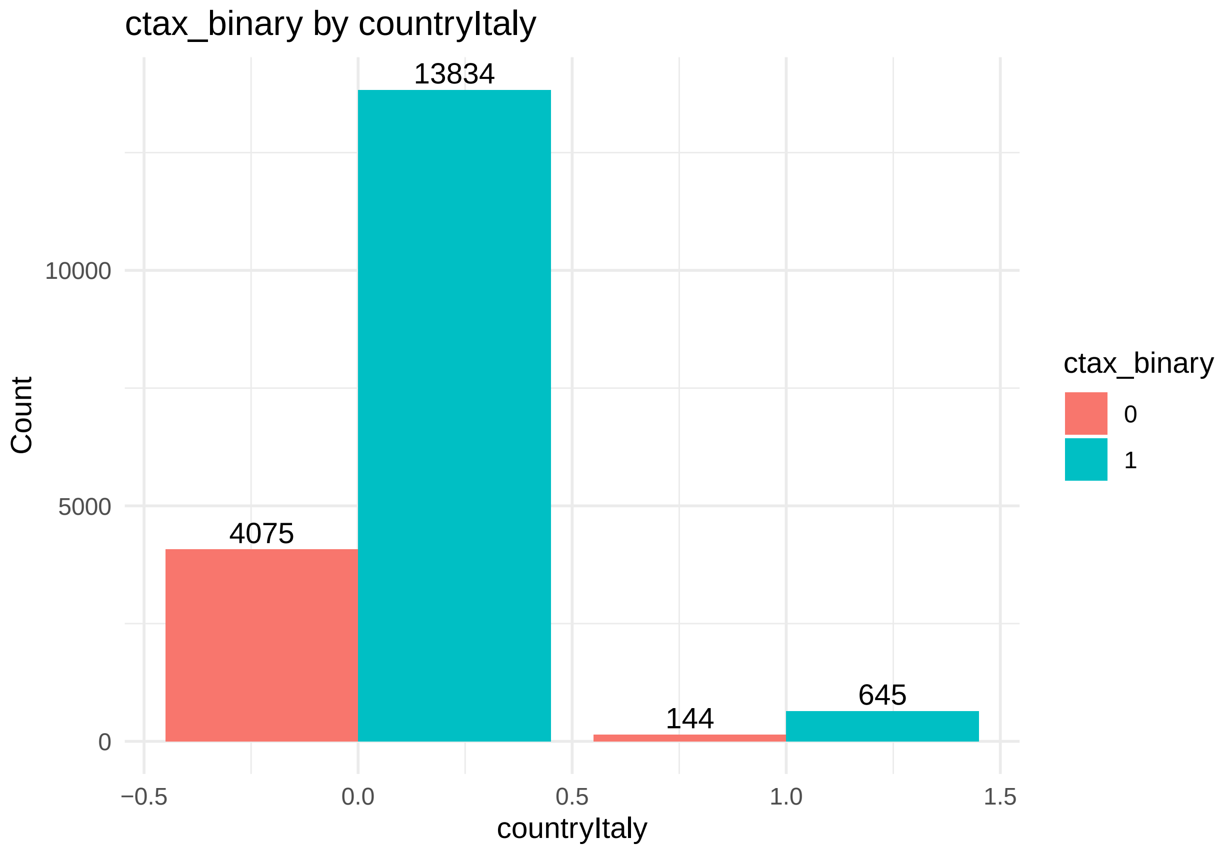


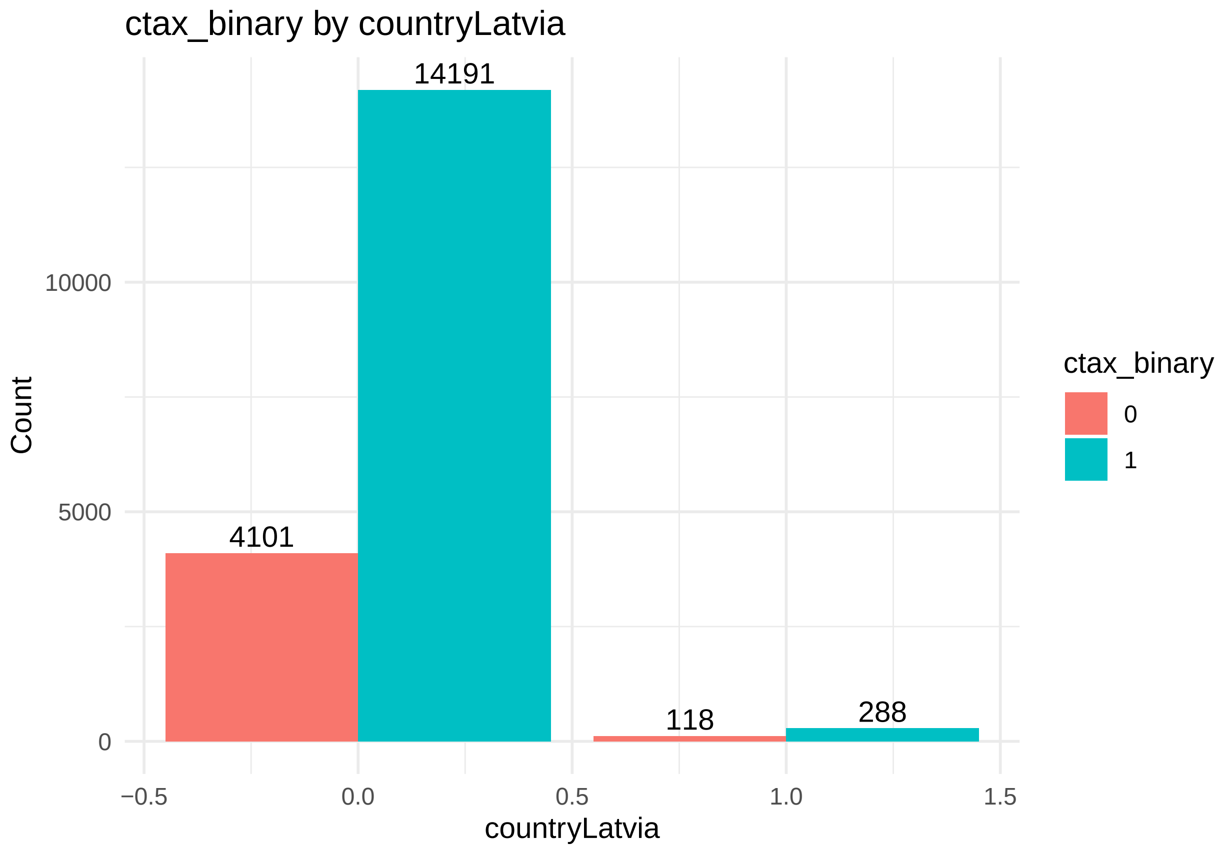
ctax_binary by countryGreece
13712
4118
101
767
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryGreece
ctax_binary
0
1

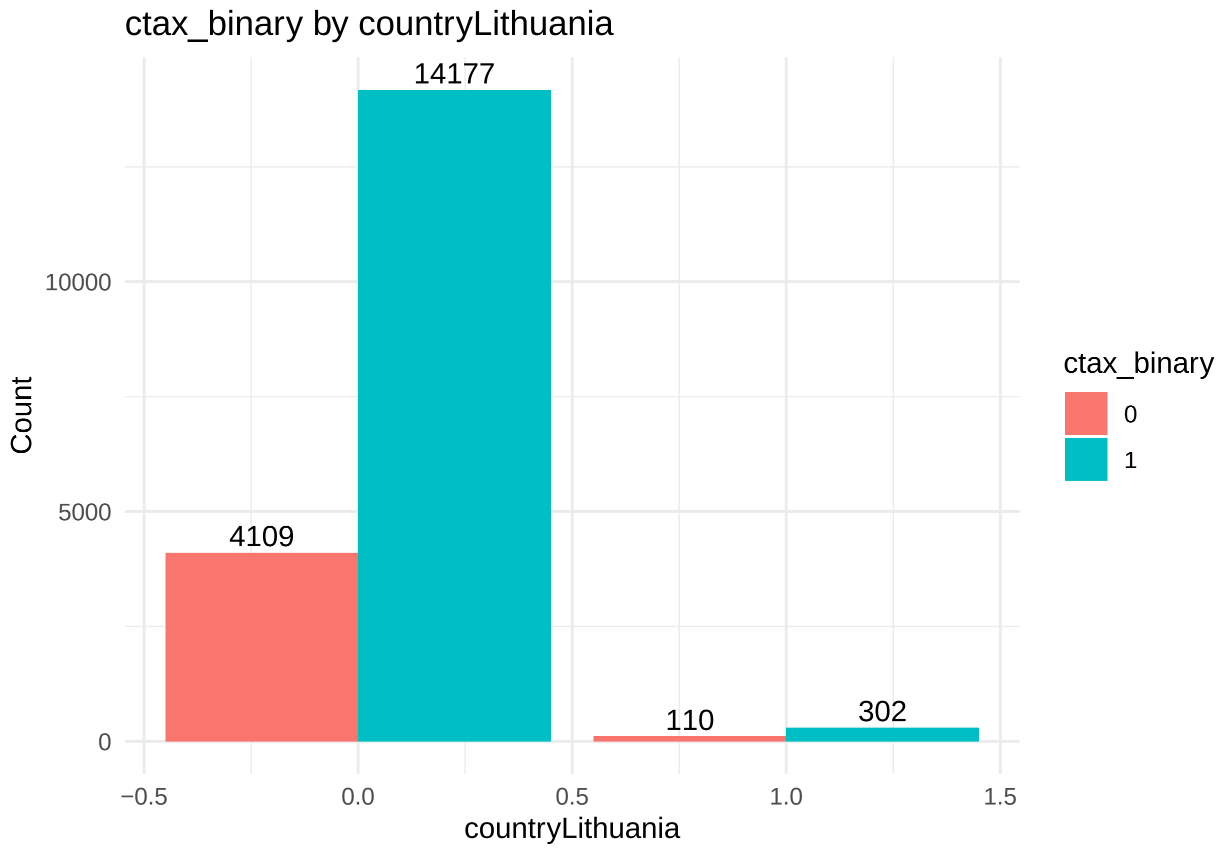

ctax_binary by countryHungary
13908
4020
199
571
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryHungary
ctax_binary
0
1



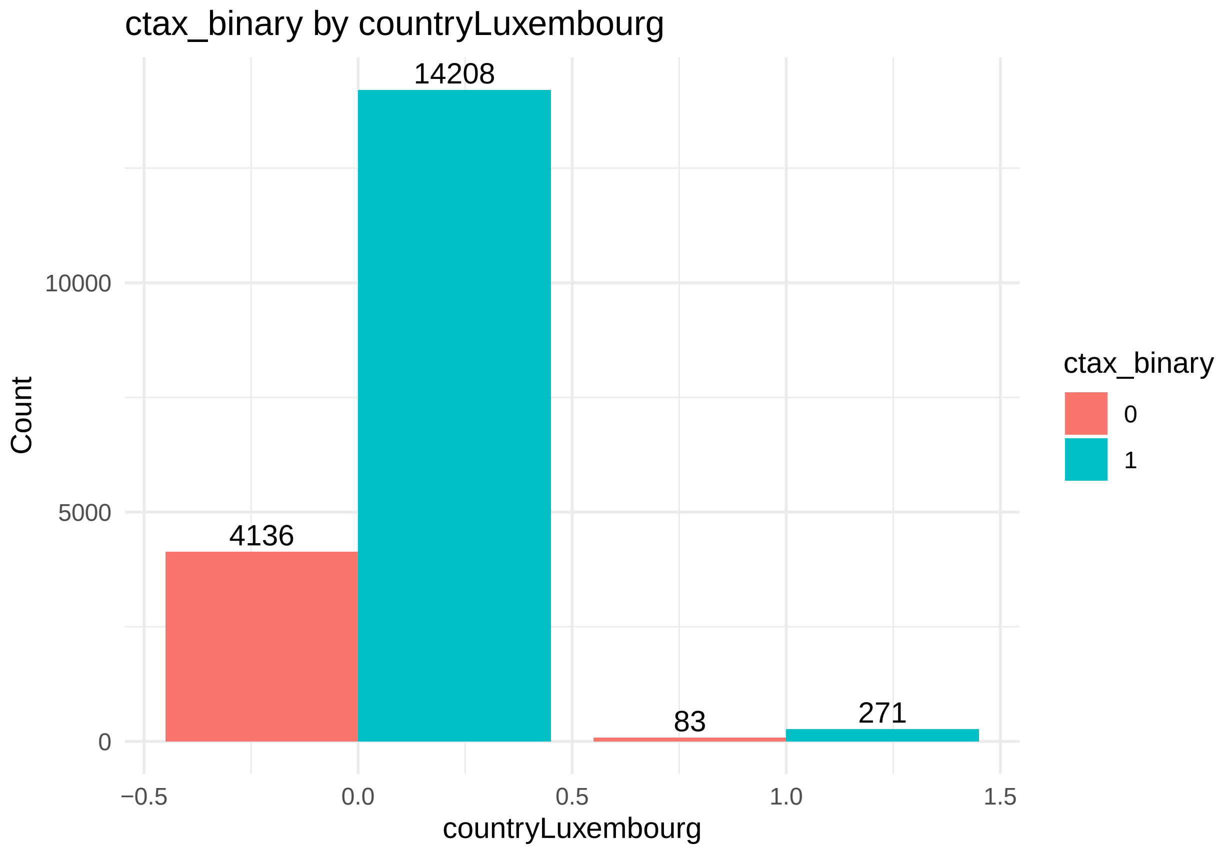
ctax_binary by countryIreland
13766
4072
147
713
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryIreland
ctax_binary
0
1

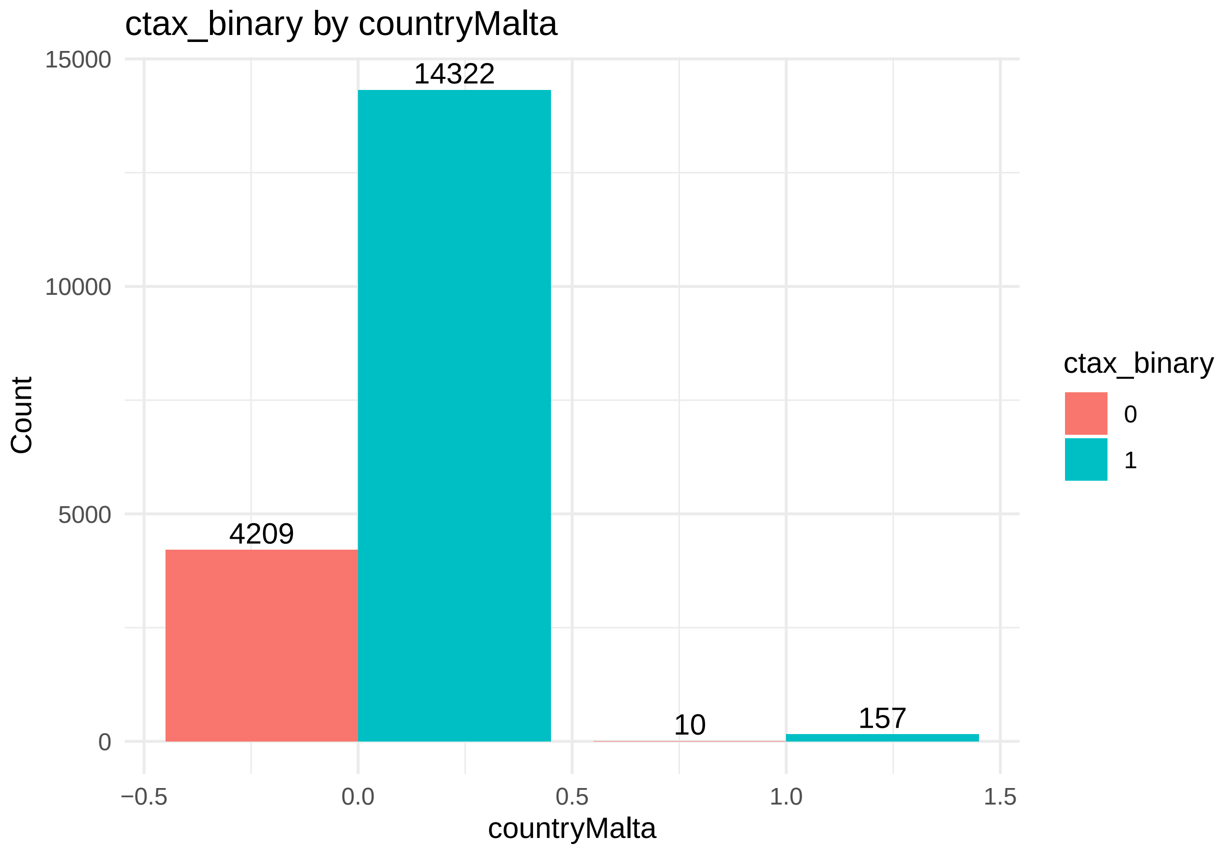

ctax_binary by countryItaly
13834
4075
144
645
Count
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countryItaly
ctax_binary
0
1

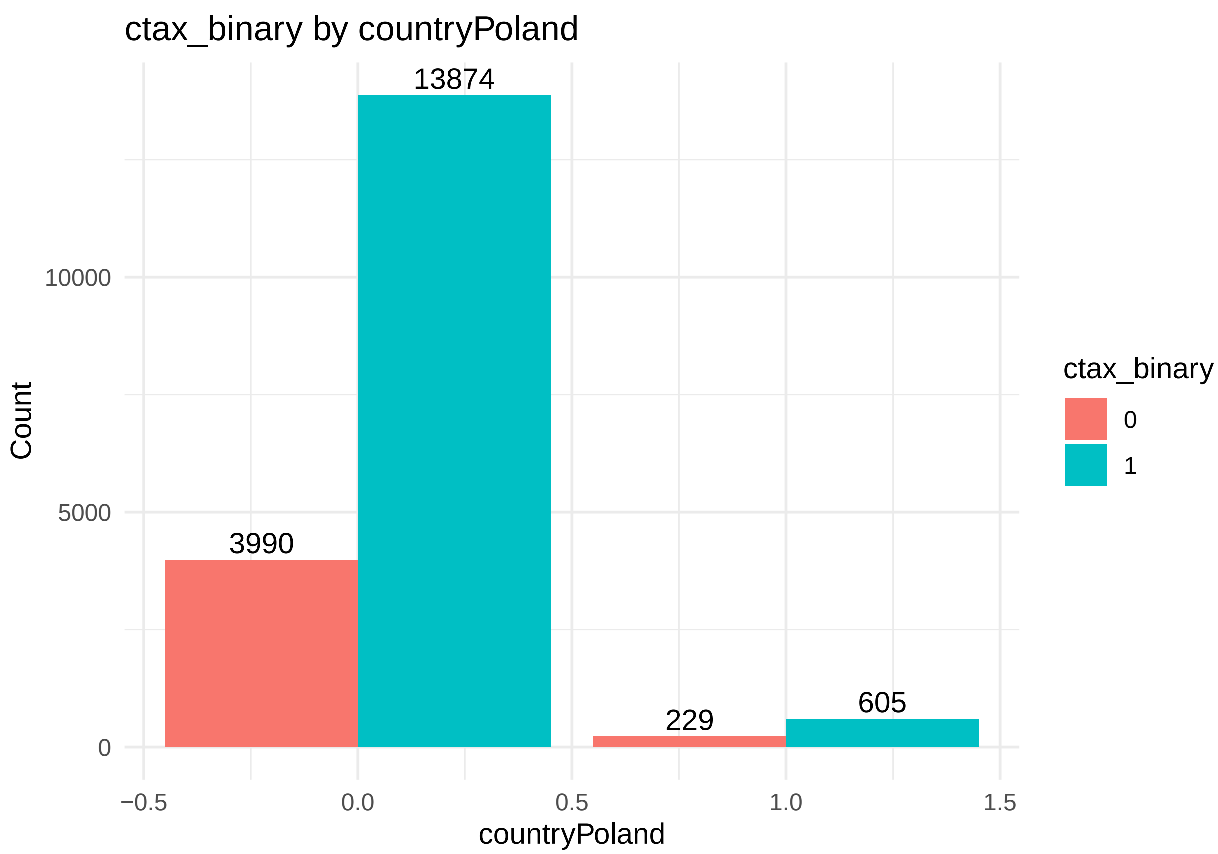

ctax_binary by countryLatvia
14191
4101
118
288
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryLatvia
ctax_binary
0
1

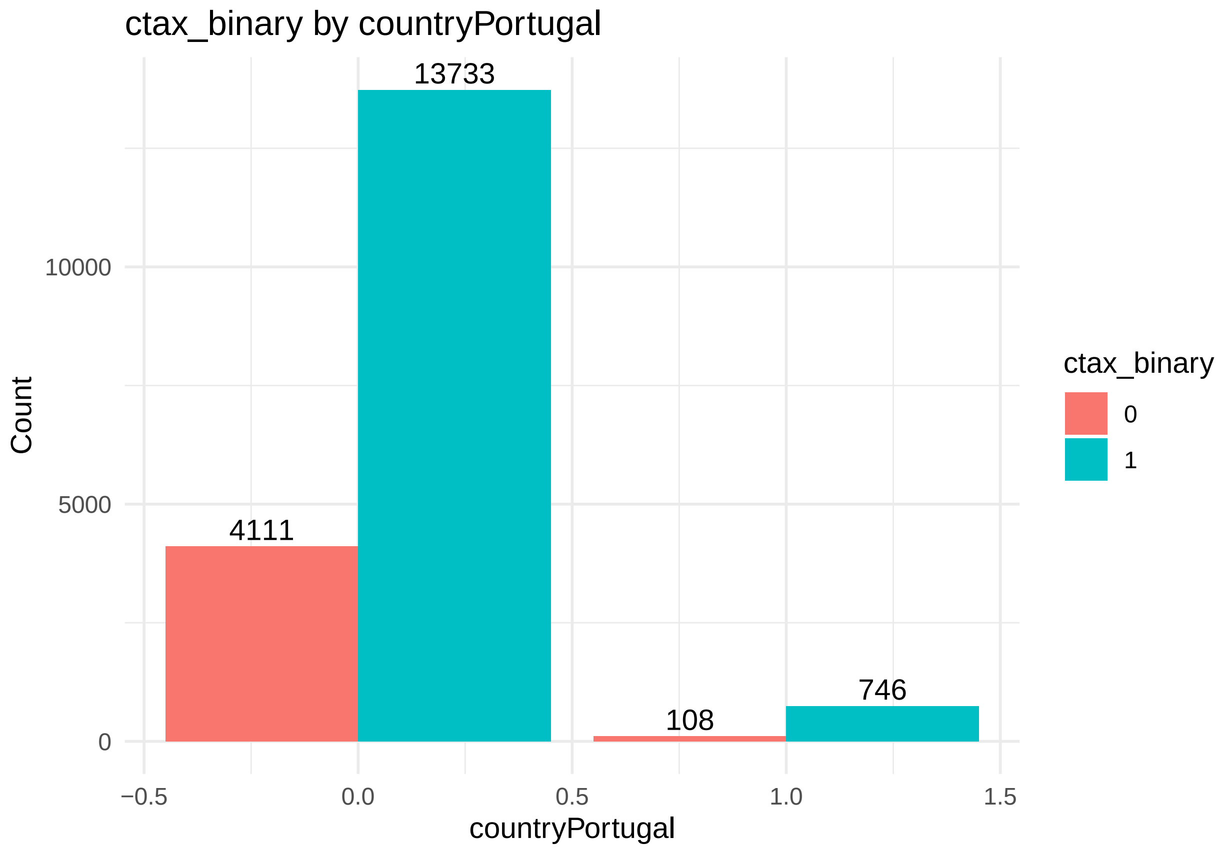


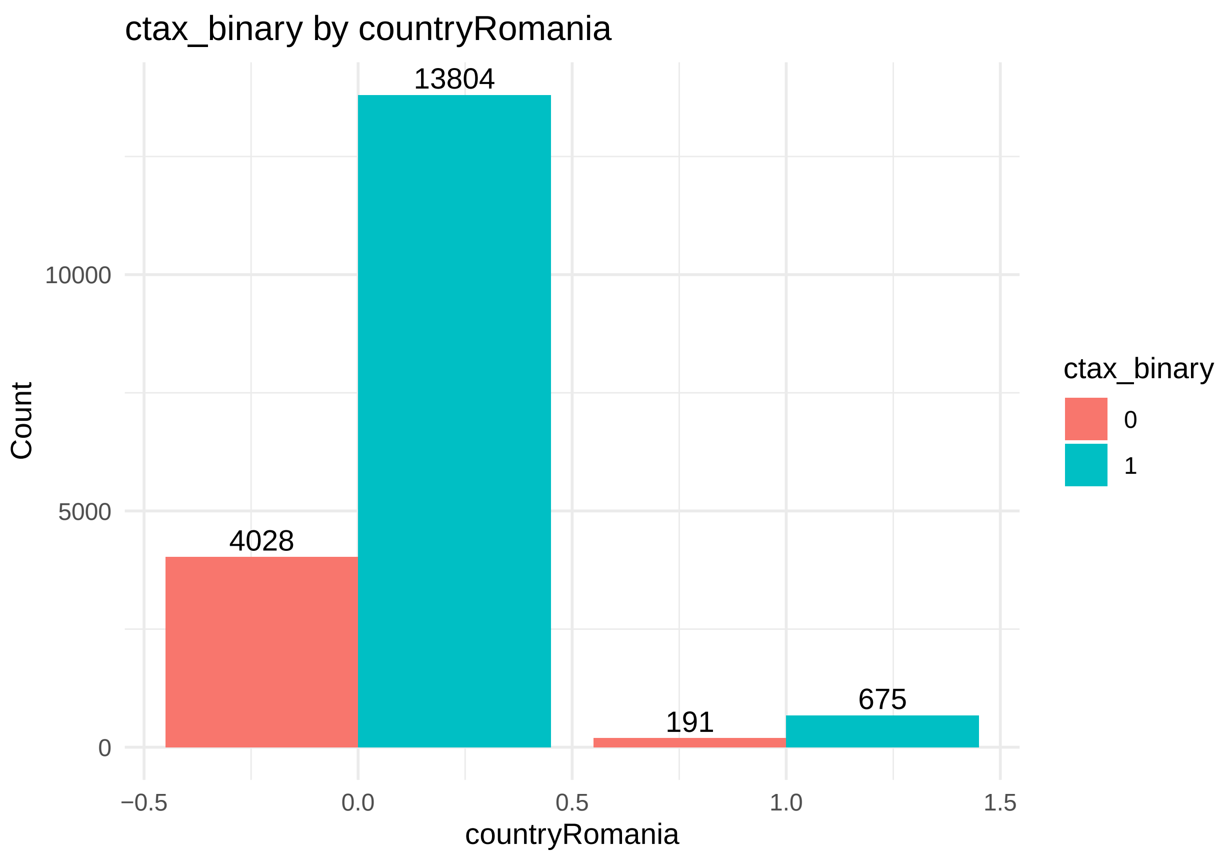
ctax_binary by countryLithuania
14177
4109
110
302
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryLithuania
ctax_binary
0
1

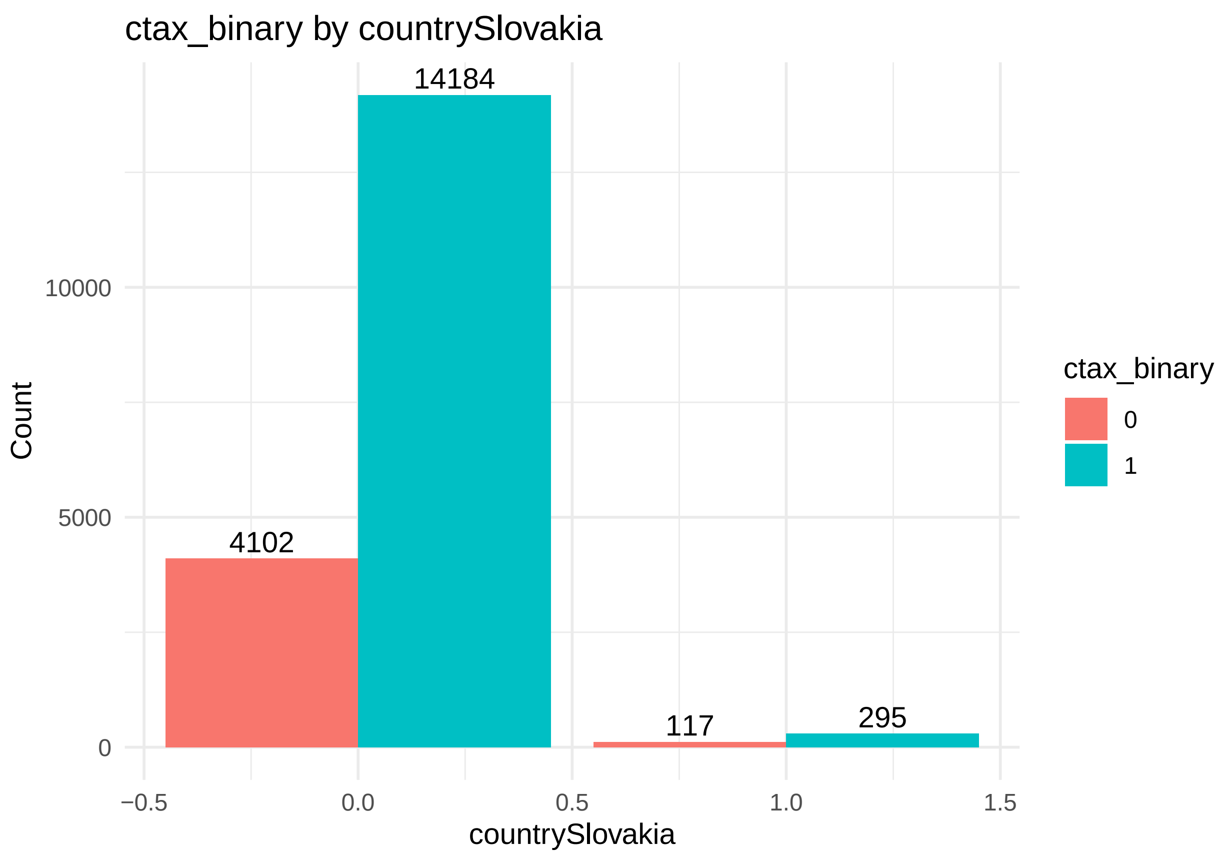

ctax_binary by countryLuxembourg
14208
4136
83
271
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryLuxembourg
ctax_binary
0
1


ctax_binary by countryMalta
15000
10000
5000
0
14322
4209
10
157
Count
−0.5
0.0
0.5
1.0
1.5
countryMalta
ctax_binary
0
1

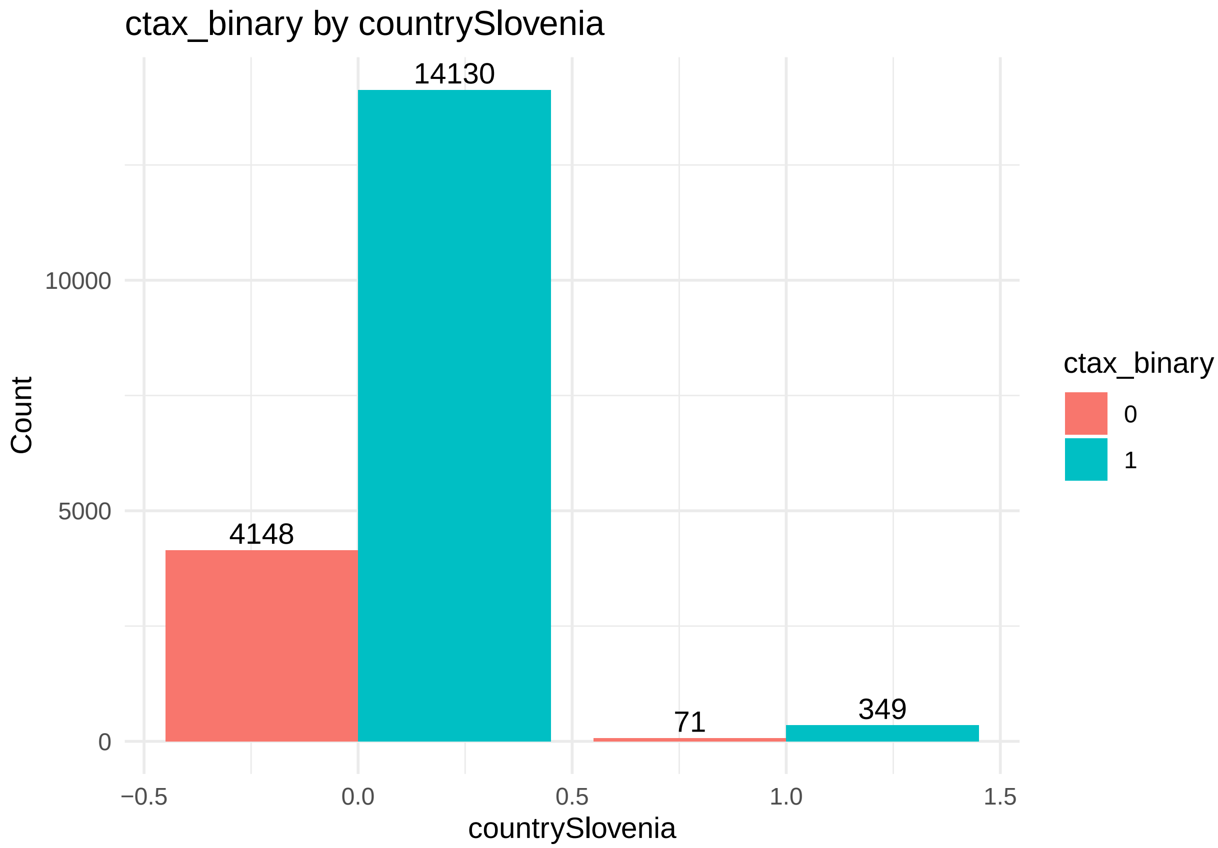

ctax_binary by countryPoland
13874
3990
229
605
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryPoland
ctax_binary
0
1


ctax_binary by countryPortugal
13733
4111
108
746
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryPortugal
ctax_binary
0
1

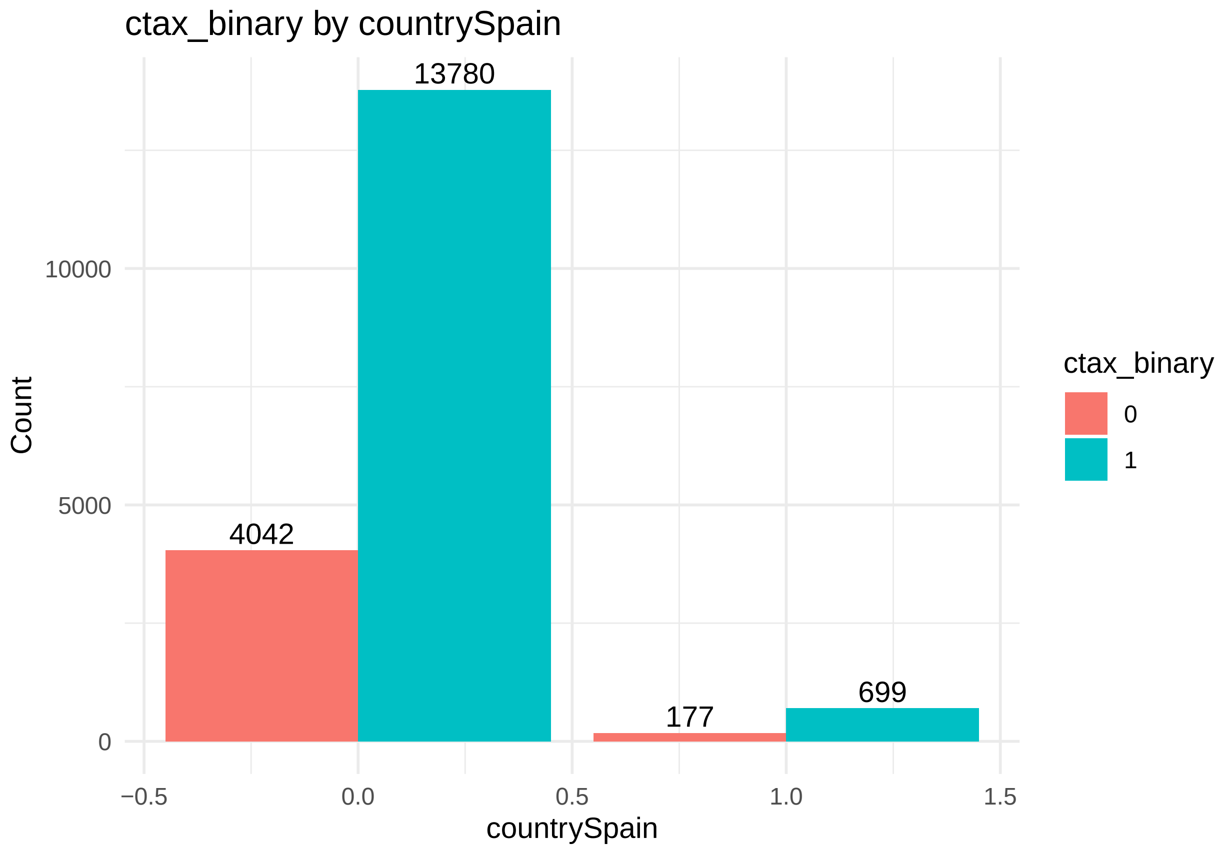

ctax_binary by countryRomania
13804
4028
191
675
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countryRomania
ctax_binary
0
1



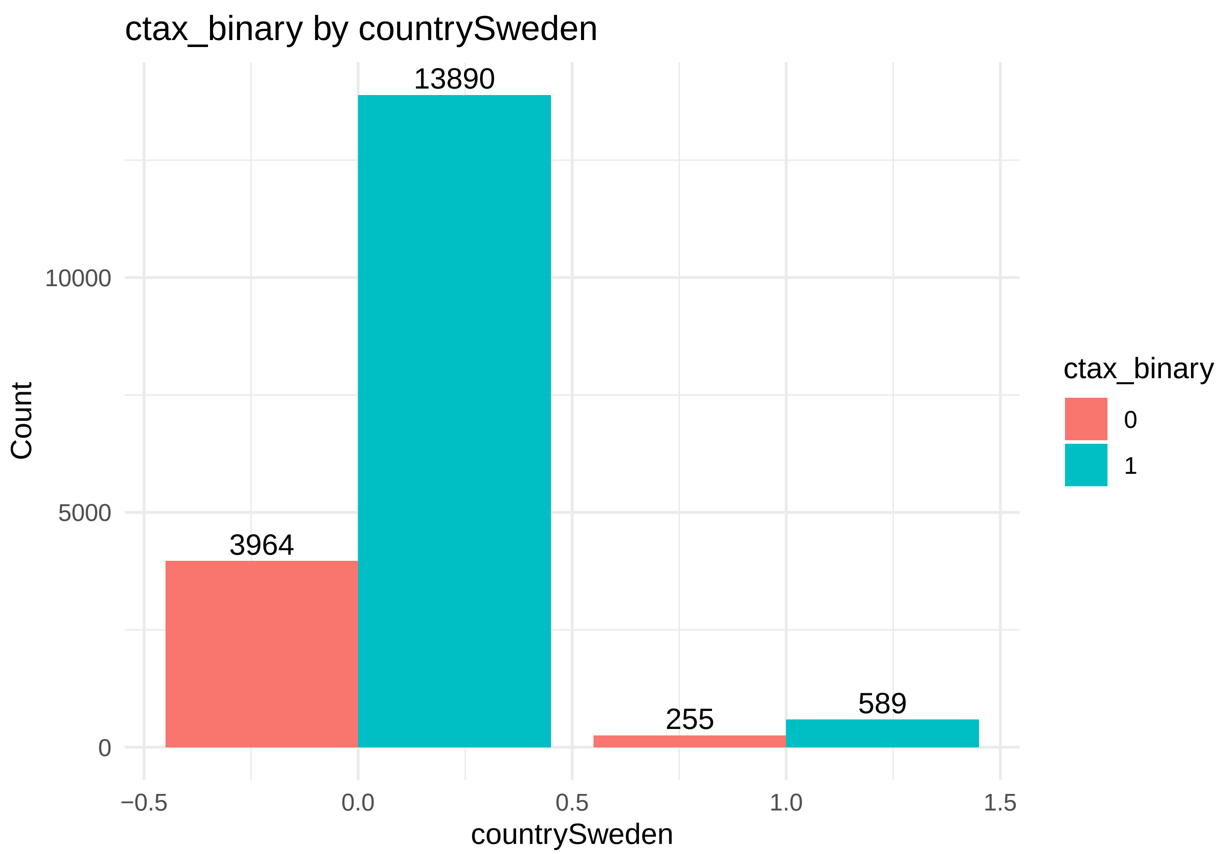
ctax_binary by countrySlovakia
14184
4102
117
295
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countrySlovakia
ctax_binary
0
1

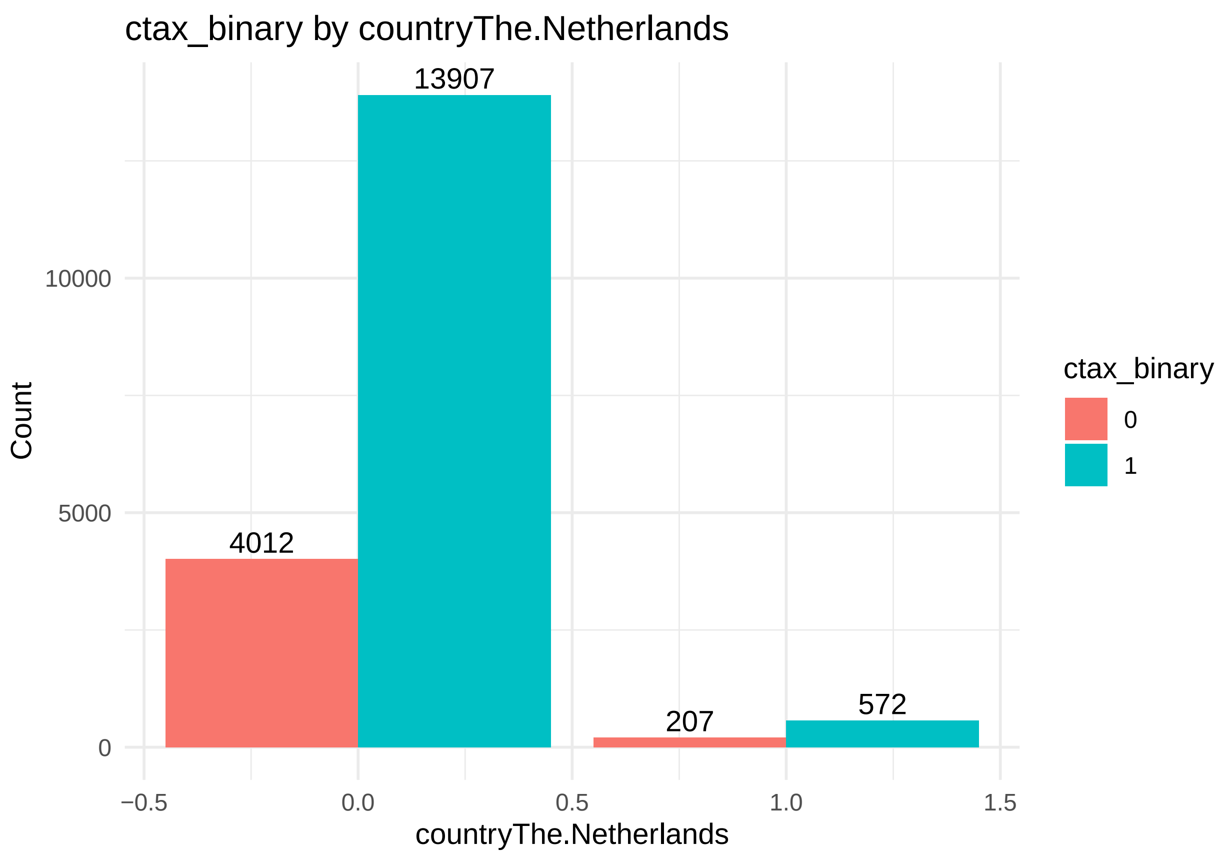


ctax_binary by countrySlovenia
14130
4148
71
349
Count
10000
5000
0
−0.5
0.0
0.5
1.0
1.5
countrySlovenia
ctax_binary
0
1



ctax_binary by countrySpain
13780
4042
177
699
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countrySpain
ctax_binary
0
1


ctax_binary by countrySweden
13890
3964
255
589
10000
5000
0
Count
−0.5
0.0
0.5
1.0
1.5
countrySweden
ctax_binary
0
1



Creating the Education Variable,Residence variable respect to education dummies and residence dummies:

```
df_clean$education <- with(df_clean, ifelse(df_clean$educationprimary == 1, "primary",
                                    ifelse(df_clean$educationtertiary == 1, "tertiary", "secondary")))
df_clean$education <- factor(df_clean$education, levels = c("secondary", "primary", "tertiary"))
df_clean$residence <- with(df_clean, ifelse(urbanizationrural == 1, "rural",
                                     ifelse(urbanizationtown == 1, "town", "city")))

df_clean$residence <- factor(df_clean$residence, levels = c("city", "town", "rural"))

df_clean=df_clean[,-c(13:17)]
```

Creating Validation Sets:

We will create 80/20 Train and Test data set.

```
set.seed(123) # For the reproducibility
train_index <- createDataPartition(df_clean$ctax_binary, p = 0.8, list = FALSE)
train_data <- df_clean[train_index, ]
test_data  <- df_clean[-train_index, ]
```

Creating the Baseline Logistic Regression Model:

Create two models : One with and one without any_cc_last2year_factor variable.

Without any_cc_last2year_factor:

```
country_vars <- names(df)[grepl("^country", names(df))]
country_vars=country_vars[-1]
train_data$ctax_binary=as.factor(train_data$ctax_binary)
```

```
paste(country_vars, collapse = " + ")
```

```
## [1] "countryBelgium + countryBulgaria + countryCroatia + countryCyprus + countryCzech.Republic + cou
```

```
full_formula=as.formula(ctax_binary~income_scale+trust+LR_scale_scale+new_ccknowledge_index_scale+resid
```

With any_cc_last2year_factor :

```
full_formula_with_cc=as.formula(ctax_binary~any_cc_last2year_factor+income_scale+trust+LR_scale_scale+n
```

```
set.seed(1)
baseline_model_with_cc=glm(formula = full_formula_with_cc,
         data = train_data, weights = country_w,family = binomial(link = "logit"),subset = regional_het
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
summary(baseline_model_with_cc)
```

```
##
## Call:
## glm(formula = full_formula_with_cc, family = binomial(link = "logit"),
##     data = train_data, weights = country_w, subset = regional_heterogeneity ==
##         1)
##
## Coefficients: (10 not defined because of singularities)
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  0.124650   0.115978   1.075  0.28248
## any_cc_last2year_factor     -0.015918   0.054307  -0.293  0.76944
## income_scale                -0.042643   0.026540  -1.607  0.10811
## trustNo really confident     0.522837   0.065005   8.043 8.77e-16 ***
## trustRather confident        1.028047   0.073736  13.942  < 2e-16 ***
## trustVery confident          1.565422   0.116714  13.412  < 2e-16 ***
## LR_scale_scale              -0.083681   0.011405  -7.337 2.18e-13 ***
## new_ccknowledge_index_scale  3.465530   0.162580  21.316  < 2e-16 ***
## residencetown               -0.057702   0.056009  -1.030  0.30290
## residencerural              -0.157699   0.068027  -2.318  0.02044 *
## age_scale                    0.001520   0.001599   0.951  0.34185
```

```
## gendermale                   -0.024335   0.050121  -0.486  0.62730
## educationprimary              0.113683   0.070946   1.602  0.10907
## educationtertiary             0.084285   0.055204   1.527  0.12681
## has_childrenyes               0.037041   0.054953   0.674  0.50028
## countryBelgium                0.368496   0.131468   2.803  0.00506 **
## countryBulgaria               0.963051   0.141415   6.810 9.75e-12 ***
## countryCroatia                1.101087   0.143966   7.648 2.04e-14 ***
## countryCyprus                       NA         NA      NA       NA
## countryCzech.Republic         0.359325   0.128755   2.791  0.00526 **
## countryDenmark                0.588647   0.136647   4.308 1.65e-05 ***
## countryEstonia                      NA         NA      NA       NA
## countryFinland                      NA         NA      NA       NA
## countryFrance                 0.568806   0.138433   4.109 3.98e-05 ***
## countryGermany                0.008216   0.124181   0.066  0.94725
## countryGreece                 1.347111   0.156113   8.629  < 2e-16 ***
## countryHungary                      NA         NA      NA       NA
## countryIreland                      NA         NA      NA       NA
## countryItaly                  0.818755   0.140729   5.818 5.96e-09 ***
## countryLatvia                       NA         NA      NA       NA
## countryLithuania                    NA         NA      NA       NA
## countryLuxembourg                   NA         NA      NA       NA
## countryMalta                        NA         NA      NA       NA
## countryPoland                 0.207412   0.128444   1.615  0.10635
## countryPortugal               1.364291   0.152059   8.972  < 2e-16 ***
## countryRomania                0.576754   0.129256   4.462 8.12e-06 ***
## countrySlovakia               0.208981   0.154742   1.351  0.17685
## countrySlovenia                     NA         NA      NA       NA
## countrySpain                  0.626163   0.134919   4.641 3.47e-06 ***
## countrySweden                 0.089265   0.126395   0.706  0.48004
## countryThe.Netherlands        0.337780   0.130960   2.579  0.00990 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 11590  on 10967  degrees of freedom
## Residual deviance: 10487  on 10937  degrees of freedom
## AIC: 10838
##
## Number of Fisher Scoring iterations: 4
```

```
yhat_baseline_train_cc=predict(baseline_model_with_cc,newdata=train_data,type="response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```
yhat_baseline_test_cc=predict(baseline_model_with_cc,newdata=test_data,type="response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
class_pred_train_cc <- ifelse(yhat_baseline_train_cc > 0.5, 1, 0)
class_pred_test_cc  <- ifelse(yhat_baseline_test_cc > 0.5, 1, 0)
```

Checking the Accuracy Metrics respect to the Train Data for the Baseline Model:

```r
confusionMatrix(as.factor(train_data$ctax_binary),as.factor(class_pred_train_cc))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0   536  2833
##          1   421 11169
##
##                Accuracy : 0.7825
##                  95% CI : (0.7758, 0.7891)
##     No Information Rate : 0.936
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1646
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.56008
##             Specificity : 0.79767
##          Pos Pred Value : 0.15910
##          Neg Pred Value : 0.96368
##              Prevalence : 0.06397
##          Detection Rate : 0.03583
##    Detection Prevalence : 0.22522
##       Balanced Accuracy : 0.67888
##
##        'Positive' Class : 0
##
```

Checking the Accuracy Metrics respect to the Test Data for the Baseline Model:

```r
confusionMatrix(as.factor(test_data$ctax_binary),as.factor(class_pred_test_cc))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  144  706
```

```
##          1   91 2798
##
##                Accuracy : 0.7868
##                  95% CI : (0.7734, 0.7999)
##     No Information Rate : 0.9371
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1852
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.61277
##             Specificity : 0.79852
##          Pos Pred Value : 0.16941
##          Neg Pred Value : 0.96850
##              Prevalence : 0.06285
##          Detection Rate : 0.03851
##    Detection Prevalence : 0.22733
##       Balanced Accuracy : 0.70564
##
##        'Positive' Class : 0
##
```

```r
set.seed(1)
baseline_model=glm(formula = full_formula,
         data = train_data, weights = country_w,family = binomial(link = "logit"))
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```r
summary(baseline_model)
```

```
##
## Call:
## glm(formula = full_formula, family = binomial(link = "logit"),
##     data = train_data, weights = country_w)
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                  0.111298   0.108305   1.028  0.30412
## income_scale                -0.042261   0.022709  -1.861  0.06275 .
## trustNo really confident     0.542622   0.055923   9.703  < 2e-16 ***
## trustRather confident        1.062523   0.062755  16.931  < 2e-16 ***
## trustVery confident          1.463011   0.096809  15.112  < 2e-16 ***
## LR_scale_scale              -0.078879   0.009956  -7.922 2.33e-15 ***
## new_ccknowledge_index_scale  3.337251   0.138800  24.044  < 2e-16 ***
## residencetown               -0.052793   0.048274  -1.094  0.27413
## residencerural              -0.180987   0.057454  -3.150  0.00163 **
## age_scale                    0.004192   0.001354   3.096  0.00196 **
## gendermale                  -0.031994   0.042786  -0.748  0.45460
## educationprimary             0.098942   0.062898   1.573  0.11570
## educationtertiary            0.089704   0.047123   1.904  0.05696 .
## has_childrenyes              0.059602   0.046833   1.273  0.20314
## countryBelgium               0.360642   0.131103   2.751  0.00594 **
```

```
## countryBulgaria            0.945159   0.140740   6.716 1.87e-11 ***
## countryCroatia             1.081013   0.140910   7.672 1.70e-14 ***
## countryCyprus              1.175089   0.215932   5.442 5.27e-08 ***
## countryCzech.Republic      0.347182   0.127715   2.718  0.00656 **
## countryDenmark             0.585273   0.135923   4.306 1.66e-05 ***
## countryEstonia            -0.030983   0.150034  -0.207  0.83640
## countryFinland             0.342999   0.129891   2.641  0.00827 **
## countryFrance              0.548620   0.136140   4.030 5.58e-05 ***
## countryGermany            -0.001514   0.121591  -0.012  0.99006
## countryGreece              1.322158   0.152765   8.655  < 2e-16 ***
## countryHungary             0.307740   0.130072   2.366  0.01799 *
## countryIreland             0.958000   0.139828   6.851 7.32e-12 ***
## countryItaly               0.801192   0.139195   5.756 8.62e-09 ***
## countryLatvia              0.205191   0.155111   1.323  0.18588
## countryLithuania           0.159447   0.156838   1.017  0.30933
## countryLuxembourg          0.465739   0.175129   2.659  0.00783 **
## countryMalta               2.225869   0.424701   5.241 1.60e-07 ***
## countryPoland              0.197634   0.127959   1.545  0.12247
## countryPortugal            1.346586   0.151540   8.886  < 2e-16 ***
## countryRomania             0.569869   0.128763   4.426 9.61e-06 ***
## countrySlovakia            0.205084   0.154375   1.328  0.18402
## countrySlovenia            0.903655   0.174709   5.172 2.31e-07 ***
## countrySpain               0.602861   0.132552   4.548 5.41e-06 ***
## countrySweden              0.081917   0.125238   0.654  0.51306
## countryThe.Netherlands     0.325172   0.130432   2.493  0.01267 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15803  on 14958  degrees of freedom
## Residual deviance: 14314  on 14919  degrees of freedom
## AIC: 14923
##
## Number of Fisher Scoring iterations: 5
```

```r
yhat_baseline_train=predict(baseline_model,newdata=train_data,type="response")
yhat_baseline_test=predict(baseline_model,newdata=test_data,type="response")

class_pred_train <- ifelse(yhat_baseline_train > 0.5, 1, 0)
class_pred_test  <- ifelse(yhat_baseline_test > 0.5, 1, 0)
```

Checking the Accuracy Metrics respect to the Train Data for the Baseline Model:

```r
confusionMatrix(as.factor(train_data$ctax_binary),as.factor(class_pred_train))
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction     0     1
##          0   432  2937
##          1   259 11331
##
##                Accuracy : 0.7863
##                  95% CI : (0.7797, 0.7929)
##     No Information Rate : 0.9538
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1474
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.62518
##             Specificity : 0.79415
##          Pos Pred Value : 0.12823
##          Neg Pred Value : 0.97765
##              Prevalence : 0.04619
##          Detection Rate : 0.02888
##    Detection Prevalence : 0.22522
##       Balanced Accuracy : 0.70967
##
##        'Positive' Class : 0
##
```

Checking the Accuracy Metrics respect to the Test Data for the Baseline Model:

```r
confusionMatrix(as.factor(test_data$ctax_binary),as.factor(class_pred_test))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  117  733
##          1   53 2836
##
##                Accuracy : 0.7898
##                  95% CI : (0.7764, 0.8027)
##     No Information Rate : 0.9545
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1662
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.68824
##             Specificity : 0.79462
##          Pos Pred Value : 0.13765
##          Neg Pred Value : 0.98165
```

```
##              Prevalence : 0.04547
##          Detection Rate : 0.03129
##    Detection Prevalence : 0.22733
##       Balanced Accuracy : 0.74143
##
##        'Positive' Class : 0
##
```

Creating the ML Models:

XGBoosting Model:

Creating the Grid Search for the XGBoosting Model:

```
xgb_grid <- expand.grid(
  nrounds = c(50,100,200),
  max_depth = c(500,700,1000),
  eta = c(0.01),
  gamma = c(0,1),
  colsample_bytree = 1,
  min_child_weight = 1,
  subsample = 1
)
```

Creating the model without cc_

Training the Model respect to Tuning Parameters

```
set.seed(123)
xgb_model <- train(
  full_formula,
  data = train_data,
  method = "xgbTree",
  trControl = trainControl(method = "cv", number = 5),
  tuneGrid = xgb_grid,
  metric = "Accuracy"
)
```

```
## [13:06:31] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:06:31] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:06:47] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:06:47] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:07:06] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:07:06] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:07:24] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:07:24] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:07:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:07:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:08:01] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:08:01] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
```

```
## [13:08:21] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:08:21] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:08:41] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:08:41] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:09:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:09:43] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:10:04] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:10:04] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:10:27] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:10:27] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:10:59] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:10:59] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:11:20] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:11:20] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:11:39] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:11:39] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:11:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:11:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:12:19] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:12:19] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:12:38] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:12:38] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:12:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:12:58] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:13:17] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:13:17] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:13:36] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:13:36] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:13:56] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:13:56] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:14:15] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:14:15] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:14:34] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:14:34] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:14:52] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:14:52] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:15:13] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:15:13] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:15:25] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:15:25] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:15:38] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:15:38] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:15:51] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:15:51] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:16:03] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:16:03] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:16:16] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
## [13:16:16] WARNING: src/c_api/c_api.cc:935: `ntree_limit` is deprecated, use `iteration_range` instea
```

```r
xgb_model$finalModel
```

```
## ##### xgb.Booster
## raw: 16 Mb
## call:
```

```
##   xgboost::xgb.train(params = list(eta = param$eta, max_depth = param$max_depth,
##     gamma = param$gamma, colsample_bytree = param$colsample_bytree,
##     min_child_weight = param$min_child_weight, subsample = param$subsample),
##     data = x, nrounds = param$nrounds, objective = "binary:logistic")
## params (as set within xgb.train):
##   eta = "0.01", max_depth = "500", gamma = "1", colsample_bytree = "1", min_child_weight = "1", subsa
## xgb.attributes:
##   niter
## callbacks:
##   cb.print.evaluation(period = print_every_n)
## # of features: 39
## niter: 200
## nfeatures : 39
## xNames : income_scale trustNo really confident trustRather confident trustVery confident LR_scale_sca
## problemType : Classification
## tuneValue :
##    nrounds max_depth  eta gamma colsample_bytree min_child_weight subsample
## 6     200       500 0.01     1                1                1         1
## obsLevels : 0 1
## param :
##  list()
```

Creating The Accuracy Metrics for the XgBoost Classification model for the Train data:

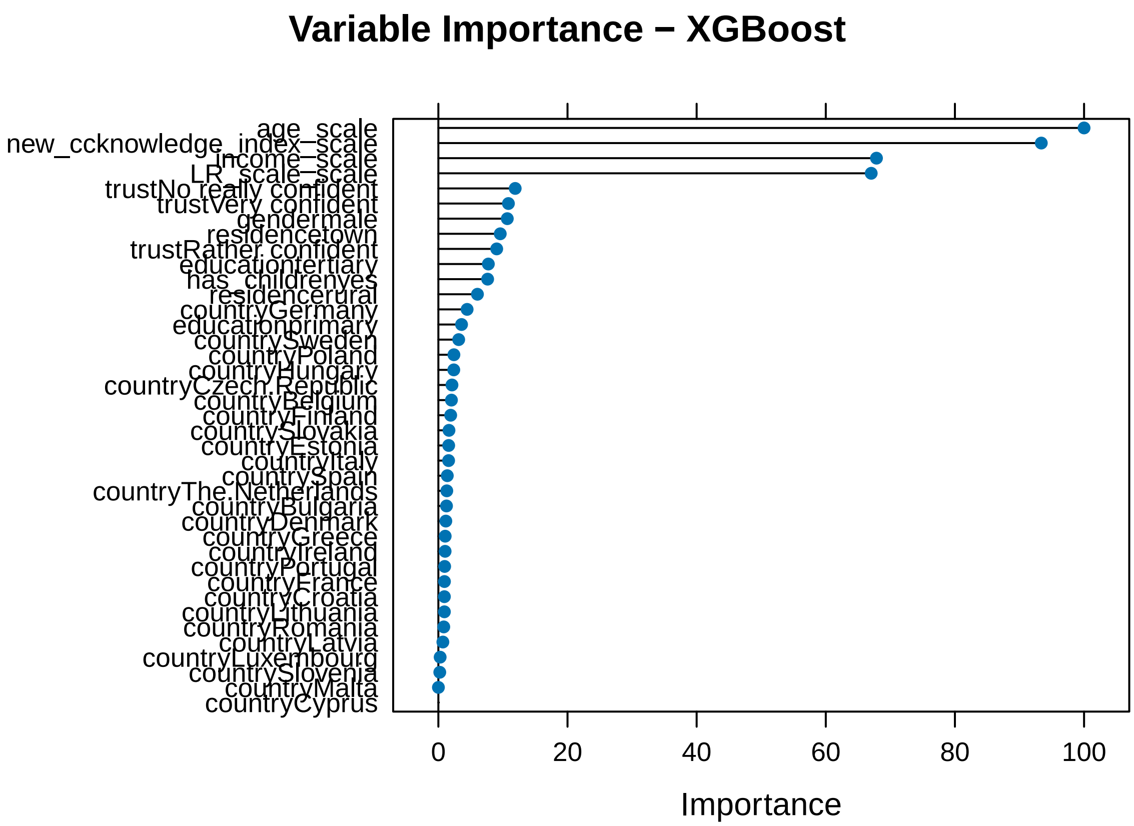
```
yhat_xgboost_train<- predict(xgb_model, newdata = train_data, type = "prob")[, "1"]
yhat_xgboost_train_binary=ifelse(yhat_xgboost_train > 0.5, 1, 0)
confusionMatrix(as.factor(train_data$ctax_binary),as.factor(yhat_xgboost_train_binary))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0  2448   921
##          1    29 11561
##
##                Accuracy : 0.9365
##                  95% CI : (0.9325, 0.9403)
##     No Information Rate : 0.8344
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7992
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9883
##             Specificity : 0.9262
##          Pos Pred Value : 0.7266
##          Neg Pred Value : 0.9975
##              Prevalence : 0.1656
##          Detection Rate : 0.1636
##    Detection Prevalence : 0.2252
##       Balanced Accuracy : 0.9573
```

```
##
##        'Positive' Class : 0
##
```

Creating Variable Importance to see which features are important for our XGBoosting model:

```r
xgb_varimp <- varImp(xgb_model)
plot(xgb_varimp, main = "Variable Importance - XGBoost")
```



Creating The Accuracy Metrics for the XgBoost Classification model for the Test data:

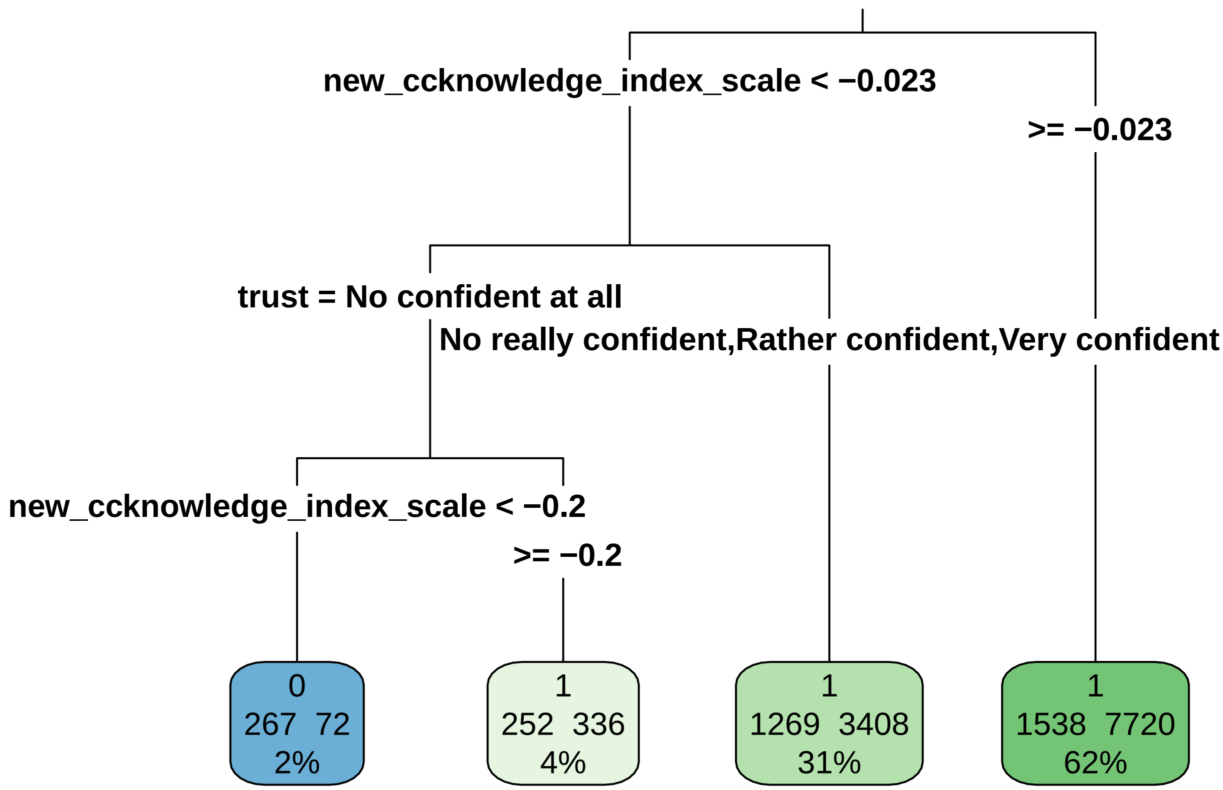
```r
yhat_xgboost_test<- predict(xgb_model, newdata = test_data, type = "prob")[, "1"]
yhat_xgboost_test_binary=ifelse(yhat_xgboost_test > 0.5, 1, 0)
confusionMatrix(as.factor(test_data$ctax_binary),as.factor(yhat_xgboost_test_binary))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  160  690
##          1  156 2733
```

```
##
##                Accuracy : 0.7737
##                  95% CI : (0.76, 0.7871)
##     No Information Rate : 0.9155
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1725
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.50633
##             Specificity : 0.79842
##          Pos Pred Value : 0.18824
##          Neg Pred Value : 0.94600
##              Prevalence : 0.08451
##          Detection Rate : 0.04279
##    Detection Prevalence : 0.22733
##       Balanced Accuracy : 0.65238
##
##        'Positive' Class : 0
##
```

Classification Tree Model:

```r
set.seed(123)
tree_model <- rpart(formula = full_formula, data = train_data, method = "class",weights =country_w )
rpart.plot(tree_model, type = 3, extra = 101, fallen.leaves = TRUE)
```

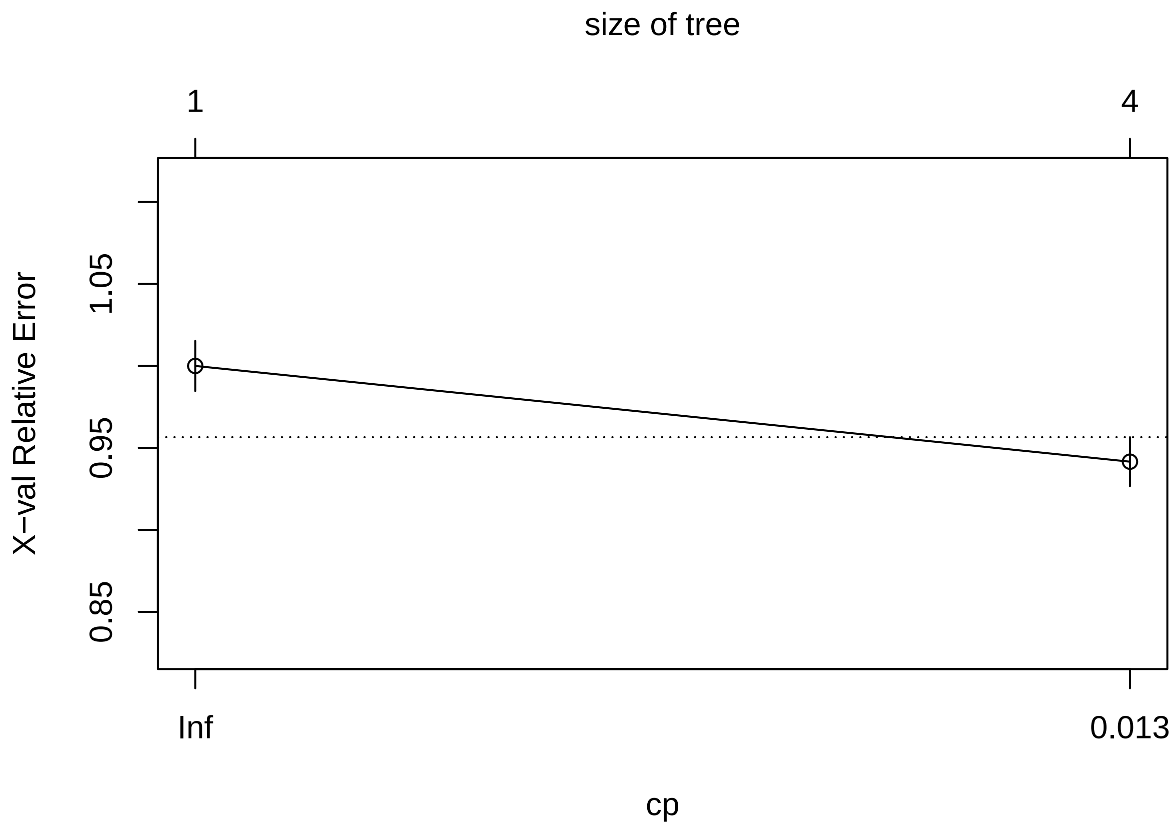


Pruning the classification tree:

```r
printcp(tree_model)
```

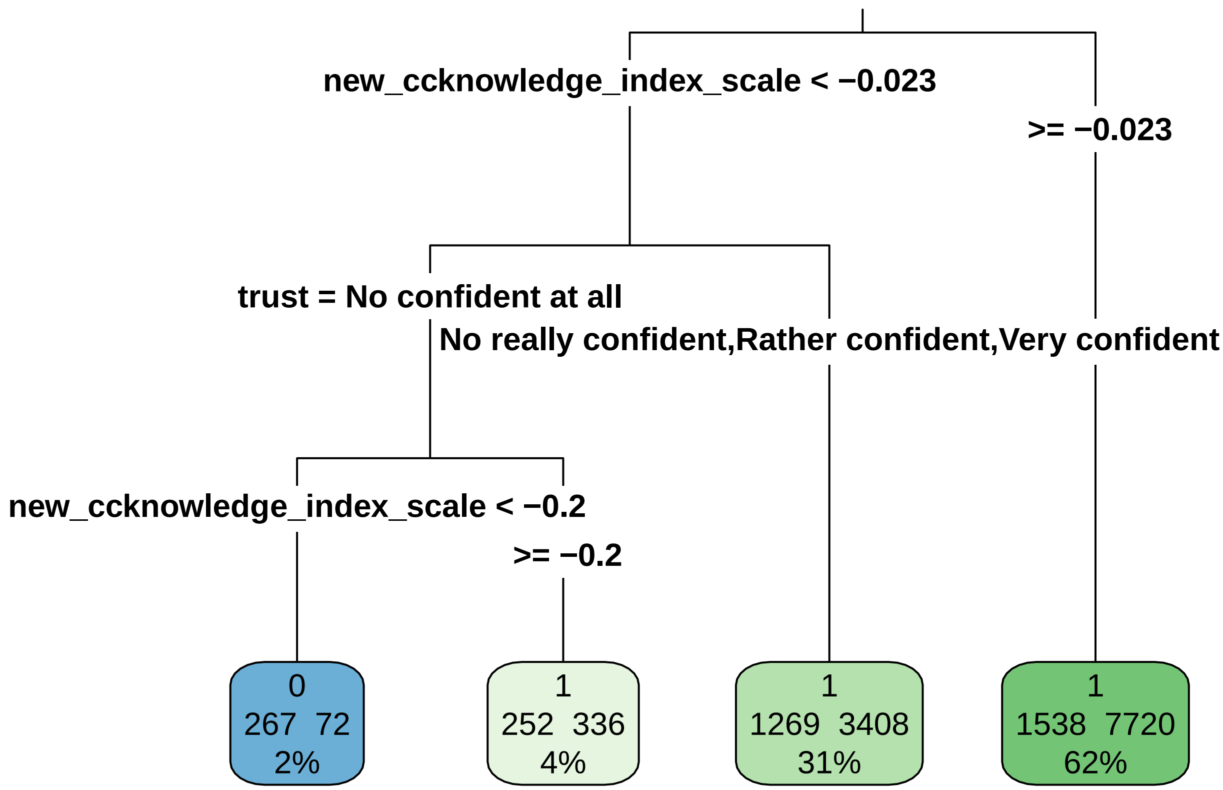
```
##
## Classification tree:
## rpart(formula = full_formula, data = train_data, weights = country_w,
##     method = "class")
##
## Variables actually used in tree construction:
## [1] new_ccknowledge_index_scale trust
##
## Root node error: 3325.8/14959 = 0.22233
##
## n= 14959
##
##          CP nsplit rel error  xerror     xstd
## 1 0.016588      0   1.00000 1.00000 0.015277
## 2 0.010000      3   0.94161 0.94161 0.014949
```

```r
plotcp(tree_model)
```




```
best_cp <- tree_model$cptable[which.min(tree_model$cptable[,"xerror"]), "CP"] #Finding the best complex
```

```
pruned_tree <- prune(tree_model, cp = best_cp)
rpart.plot(pruned_tree, type = 3, extra = 101, fallen.leaves = TRUE)
```



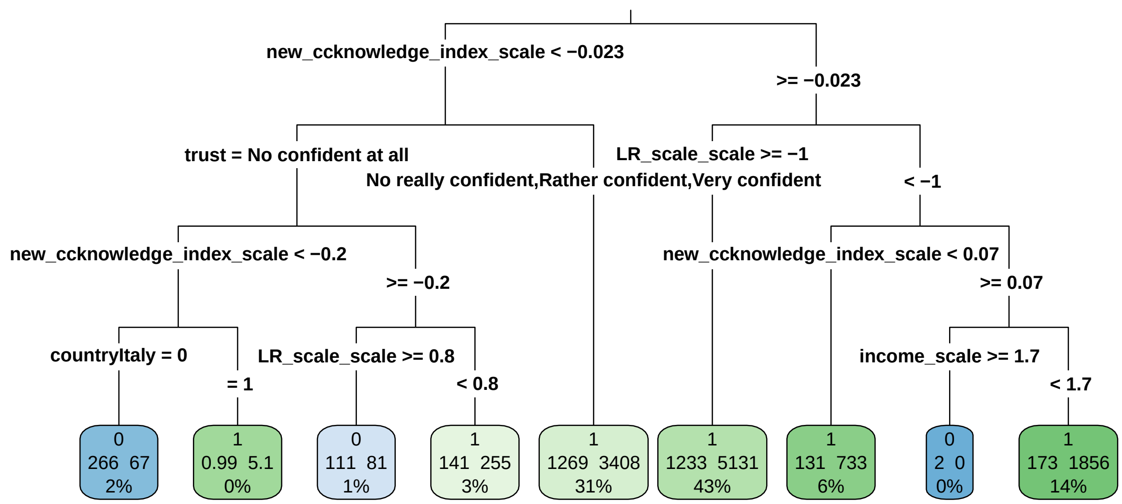
Grid Search for hyperparameter tuning in classification tree:

```
minsplit_vals <- c(10, 20, 30)
maxdepth_vals <- c(2, 4, 6)

best_model <- NULL
best_error <- Inf

for (minsplit in minsplit_vals) {
  for (maxdepth in maxdepth_vals) {
    model <- rpart(formula=full_formula, data = train_data, method = "class",weights=country_w,
                   control = rpart.control(minsplit = minsplit, maxdepth = maxdepth, cp = 0))

    # xerror from cross-validated tree
    err <- model$cptable[which.min(model$cptable[,"xerror"]), "xerror"]

    cat("minsplit =", minsplit, ", maxdepth =", maxdepth, " -> xerror =", err, "\n")

    if (err < best_error) {
      best_error <- err
      best_model <- model
    }
  }
}
```

```
## minsplit = 10 , maxdepth = 2  -> xerror = 0.9727628
```

```
## minsplit = 10 , maxdepth = 4  -> xerror = 0.9355584
## minsplit = 10 , maxdepth = 6  -> xerror = 0.9359682
## minsplit = 20 , maxdepth = 2  -> xerror = 0.9715462
## minsplit = 20 , maxdepth = 4  -> xerror = 0.9416051
## minsplit = 20 , maxdepth = 6  -> xerror = 0.9360847
## minsplit = 30 , maxdepth = 2  -> xerror = 0.9681962
## minsplit = 30 , maxdepth = 4  -> xerror = 0.9420926
## minsplit = 30 , maxdepth = 6  -> xerror = 0.9366461
```

```r
# Plot best tree
rpart.plot(best_model, type = 3, extra = 101, fallen.leaves = TRUE)
```



Creating The Accuracy Metrics for the Pruned Classification Tree model for the Train data:

```r
yhat_tree_pruned_train <- predict(pruned_tree, newdata = train_data, type = "class")
confusionMatrix(as.factor(train_data$ctax_binary),yhat_tree_pruned_train)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0   270  3099
##          1    73 11517
```

```
##
##                Accuracy : 0.788
##                  95% CI : (0.7813, 0.7945)
##     No Information Rate : 0.9771
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1084
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.78717
##             Specificity : 0.78797
##          Pos Pred Value : 0.08014
##          Neg Pred Value : 0.99370
##              Prevalence : 0.02293
##          Detection Rate : 0.01805
##    Detection Prevalence : 0.22522
##       Balanced Accuracy : 0.78757
##
##        'Positive' Class : 0
##
```

Creating The Accuracy Metrics for the Pruned Classification Tree model for the Test data:

```r
yhat_tree_pruned_test <- predict(pruned_tree, newdata = test_data, type = "class")
confusionMatrix(as.factor(test_data$ctax_binary),yhat_tree_pruned_test)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0   77  773
##          1   23 2866
##
##                Accuracy : 0.7871
##                  95% CI : (0.7736, 0.8001)
##     No Information Rate : 0.9733
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.12
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.77000
##             Specificity : 0.78758
##          Pos Pred Value : 0.09059
##          Neg Pred Value : 0.99204
##              Prevalence : 0.02675
##          Detection Rate : 0.02059
##    Detection Prevalence : 0.22733
##       Balanced Accuracy : 0.77879
##
```

```
##        'Positive' Class : 0
##
```

Creating The Accuracy Metrics for the Tuned Classification Tree model for the Train data:

```
yhat_tree_tuned_train <- predict(best_model, newdata = train_data, type = "class")
confusionMatrix(as.factor(train_data$ctax_binary),yhat_tree_tuned_train)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0   385  2984
##          1   150 11440
##
##                Accuracy : 0.7905
##                  95% CI : (0.7839, 0.797)
##     No Information Rate : 0.9642
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1444
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.71963
##             Specificity : 0.79312
##          Pos Pred Value : 0.11428
##          Neg Pred Value : 0.98706
##              Prevalence : 0.03576
##          Detection Rate : 0.02574
##    Detection Prevalence : 0.22522
##       Balanced Accuracy : 0.75637
##
##        'Positive' Class : 0
##
```

Creating The Accuracy Metrics for the Tuned Classification Tree model for the Test data:

```
yhat_tree_tuned_test <- predict(best_model, newdata = test_data, type = "class")
confusionMatrix(as.factor(test_data$ctax_binary),yhat_tree_tuned_test)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0   98  752
##          1   42 2847
##
##                Accuracy : 0.7876
```

```
##                  95% CI : (0.7742, 0.8007)
##     No Information Rate : 0.9626
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1429
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.70000
##             Specificity : 0.79105
##          Pos Pred Value : 0.11529
##          Neg Pred Value : 0.98546
##              Prevalence : 0.03744
##          Detection Rate : 0.02621
##    Detection Prevalence : 0.22733
##       Balanced Accuracy : 0.74553
##
##        'Positive' Class : 0
##
```

RandomForest Model:

Creating Grid Search Parameters for the Random Forest Model

```r
mtry_grid <- c(2, 4, 6)
ntree_grid <- c(100, 300, 500)
nodesize_grid <- c(1, 5)

best_oob <- Inf
best_model <- NULL
results <- data.frame()
```

Training the Model respect to Grid Search

```r
set.seed(123)

# Grid search loop
for (m in mtry_grid) {
  for (n in ntree_grid) {
    for (node in nodesize_grid) {

      model <- randomForest(
        full_formula,
        data = train_data,
        mtry = m,
        ntree = n,
        nodesize = node,

      )

      oob_err <- tail(model$err.rate[, "OOB"], 1)
```

```r
      # Store results
      results <- rbind(results, data.frame(mtry = m, ntree = n, nodesize = node, OOB_Error = oob_err))

      cat("mtry =", m, "| ntree =", n, "| nodesize =", node, "| OOB error =", oob_err, "\n")

      if (oob_err < best_oob) {
        best_oob <- oob_err
        best_model <- model
      }
    }
  }
}
```

```
## mtry = 2 | ntree = 100 | nodesize = 1 | OOB error = 0.2236781
## mtry = 2 | ntree = 100 | nodesize = 5 | OOB error = 0.2228759
## mtry = 2 | ntree = 300 | nodesize = 1 | OOB error = 0.2245471
## mtry = 2 | ntree = 300 | nodesize = 5 | OOB error = 0.2246139
## mtry = 2 | ntree = 500 | nodesize = 1 | OOB error = 0.2246139
## mtry = 2 | ntree = 500 | nodesize = 5 | OOB error = 0.2250819
## mtry = 4 | ntree = 100 | nodesize = 1 | OOB error = 0.2132495
## mtry = 4 | ntree = 100 | nodesize = 5 | OOB error = 0.2133164
## mtry = 4 | ntree = 300 | nodesize = 1 | OOB error = 0.2119126
## mtry = 4 | ntree = 300 | nodesize = 5 | OOB error = 0.2120463
## mtry = 4 | ntree = 500 | nodesize = 1 | OOB error = 0.2115783
## mtry = 4 | ntree = 500 | nodesize = 5 | OOB error = 0.2111772
## mtry = 6 | ntree = 100 | nodesize = 1 | OOB error = 0.2133164
## mtry = 6 | ntree = 100 | nodesize = 5 | OOB error = 0.2133832
## mtry = 6 | ntree = 300 | nodesize = 1 | OOB error = 0.2101745
## mtry = 6 | ntree = 300 | nodesize = 5 | OOB error = 0.2109098
## mtry = 6 | ntree = 500 | nodesize = 1 | OOB error = 0.2113778
## mtry = 6 | ntree = 500 | nodesize = 5 | OOB error = 0.2119794
```

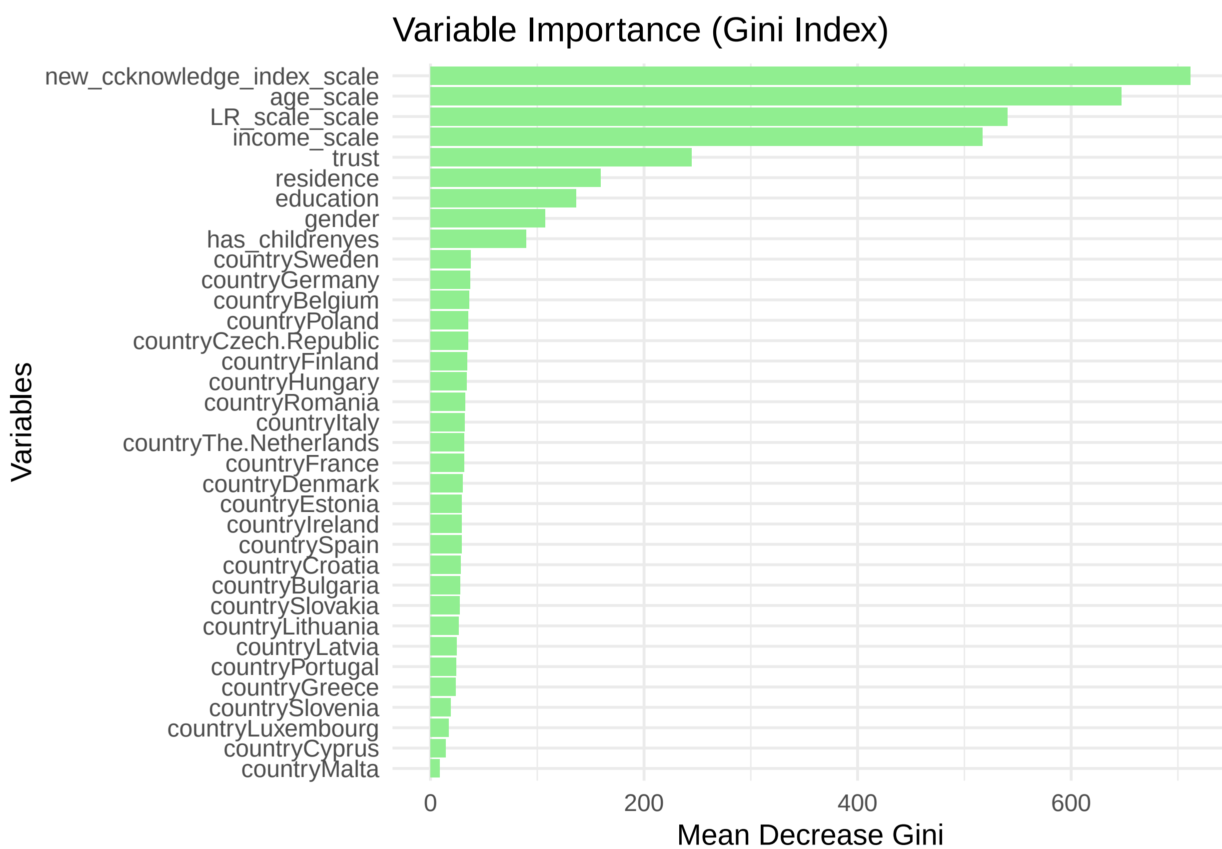
```r
best_model
```

```
##
## Call:
##  randomForest(formula = full_formula, data = train_data, mtry = m,      ntree = n, nodesize = node, 
##                Type of random forest: classification
##                      Number of trees: 300
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 21.02%
## Confusion matrix:
##     0     1 class.error
## 0 475  2894  0.85900861
## 1 250 11340  0.02157032
```

Creating Variable Importance to see which features are important for our randomForest model:

```r
imp <-importance(best_model, type = 2) # Getting the feature importance of the best model respect to Me
imp_df <- data.frame(Variable = rownames(imp), Importance = imp[, 1])
```

```r
# Plot using ggplot2
ggplot(imp_df, aes(x = reorder(Variable, Importance), y = Importance)) +
  geom_col(fill = "lightgreen") +
  coord_flip() +
  labs(title = "Variable Importance (Gini Index)",
       x = "Variables", y = "Mean Decrease Gini") +
  theme_minimal()
```



Creating The Accuracy Metrics for the RandomForest model for the Train data:

```r
yhat_rf_train=predict(best_model, newdata = train_data, type = "response")
confusionMatrix(as.factor(train_data$ctax_binary),yhat_rf_train)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0  2625   744
##          1     2 11588
##
##                Accuracy : 0.9501
##                  95% CI : (0.9465, 0.9536)
```

```
##     No Information Rate : 0.8244
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.845
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9992
##             Specificity : 0.9397
##          Pos Pred Value : 0.7792
##          Neg Pred Value : 0.9998
##              Prevalence : 0.1756
##          Detection Rate : 0.1755
##    Detection Prevalence : 0.2252
##       Balanced Accuracy : 0.9695
##
##        'Positive' Class : 0
##
```

Creating The Accuracy Metrics for the RandomForest model for the Test data:

```r
yhat_rf_test= predict(best_model, newdata = test_data, type = "response")
confusionMatrix(as.factor(test_data$ctax_binary),yhat_rf_test)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  123  727
##          1   52 2837
##
##                Accuracy : 0.7917
##                  95% CI : (0.7783, 0.8046)
##     No Information Rate : 0.9532
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.176
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7029
##             Specificity : 0.7960
##          Pos Pred Value : 0.1447
##          Neg Pred Value : 0.9820
##              Prevalence : 0.0468
##          Detection Rate : 0.0329
##    Detection Prevalence : 0.2273
##       Balanced Accuracy : 0.7494
##
##        'Positive' Class : 0
##
```

```r
table(test_data$ctax_binary,yhat_rf_test)
```

```
##    yhat_rf_test
##        0    1
##   0  123  727
##   1   52 2837
```

Starting SRM:

Defining the smoothed residual function:

```r
compute_smoothed_residuals=function(yhat_ml,yhat_baseline,y_true){
  sm_res=((y_true-yhat_baseline)^2) +
         (2 * (y_true - yhat_baseline) * (yhat_ml - y_true))+
          ((yhat_ml - y_true)^2)
  return(sm_res)
}
```

```r
yhat_baseline=predict(baseline_model,newdata=test_data,type="response")
yhat_rf= predict(best_model, newdata = test_data, type = "prob")[, "1"]
```

Creating the Smoothed Residuals:

```r
srm_df=test_data
srm_df["Smoothed_Residuals"]=compute_smoothed_residuals(yhat_ml=yhat_rf,yhat_baseline,test_data$ctax_bin
```

Finding top 100 smoothed residuals respect to the RandomForest and baseline Logit model:

```r
srm_df_sorted <- srm_df %>% arrange(desc(Smoothed_Residuals))
srm_final=head(srm_df_sorted,100)
head(srm_final)
```

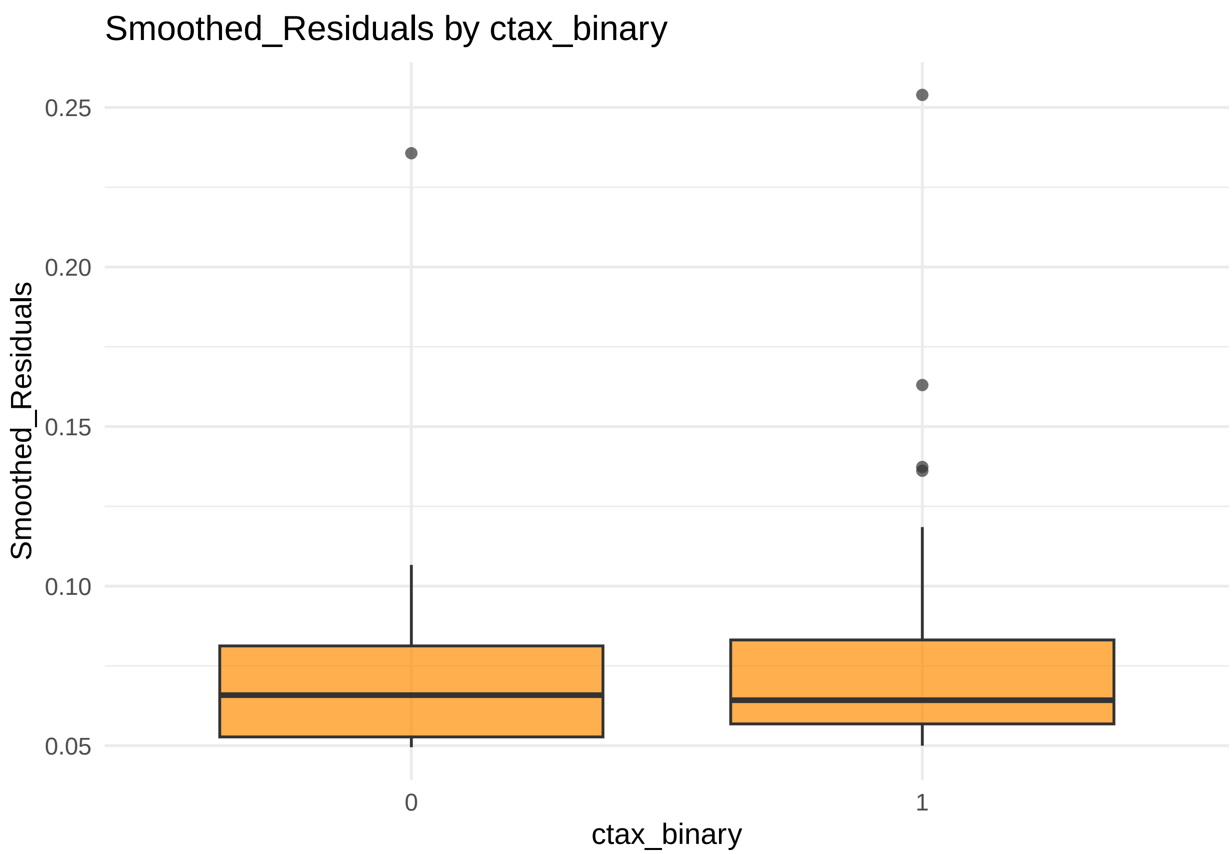
```
##   ctax_binary income_scale age_scale               trust LR_scale_scale
## 1           1   1.49539445     9.664 No really confident    -0.01511879
## 2           0   1.33867734    -5.537      Very confident     3.94488189
## 3           1   0.29317216   -13.714 No really confident     3.59303591
## 4           1   0.37555168   -24.474    Rather confident     2.57435345
## 5           1  -1.28288379   -11.778 No really confident     1.66515837
## 6           1  -0.01744122   -21.233    Rather confident     2.36582694
##   new_ccknowledge_index_scale any_cc_last2year_factor regional_heterogeneity
## 1                  0.13633333                       1                      0
## 2                  0.05433333                       1                      1
## 3                 -0.15372222                       0                      1
## 4                 -0.39711111                       1                      1
## 5                 -0.28488889                       1                      1
## 6                 -0.30327778                       0                      1
##   LR_scale new_ccknowledge_index gender country_w has_childrenyes
## 1        5             0.7777778   male  0.800470               0
## 2        9             0.7222222   male  1.172674               1
## 3        9             0.5000000   male  0.970778               1
```

```
## 4       8            0.2777778 female  0.971153              1
## 5       7            0.3333333   male  1.152172              0
## 6       8            0.3888889   male  0.747723              1
##   countryBelgium countryBulgaria countryCroatia countryCyprus
## 1              0               0              0             0
## 2              0               0              1             0
## 3              0               0              0             0
## 4              0               0              0             0
## 5              0               0              0             0
## 6              0               0              0             0
##   countryCzech.Republic countryDenmark countryEstonia countryFinland
## 1                     0              0              0              0
## 2                     0              0              0              0
## 3                     0              0              0              0
## 4                     0              0              0              0
## 5                     0              0              0              0
## 6                     0              0              0              0
##   countryFrance countryGermany countryGreece countryHungary countryIreland
## 1             0              0             0              0              0
## 2             0              0             0              0              0
## 3             0              0             0              0              0
## 4             0              1             0              0              0
## 5             0              0             0              0              0
## 6             0              0             0              0              0
##   countryItaly countryLatvia countryLithuania countryLuxembourg countryMalta
## 1            0             0                0                 0            0
## 2            0             0                0                 0            0
## 3            0             0                0                 0            0
## 4            0             0                0                 0            0
## 5            0             0                0                 0            0
## 6            0             0                0                 0            0
##   countryPoland countryPortugal countryRomania countrySlovakia countrySlovenia
## 1             0               0              0               0               1
## 2             0               0              0               0               0
## 3             0               0              0               0               0
## 4             0               0              0               0               0
## 5             0               0              0               1               0
## 6             0               0              0               0               0
##   countrySpain countrySweden countryThe.Netherlands education residence
## 1            0             0                      0 secondary      town
## 2            0             0                      0  tertiary     rural
## 3            0             0                      0  tertiary      city
## 4            0             0                      0  tertiary      city
## 5            0             0                      0   primary     rural
## 6            0             1                      0 secondary      city
##   Smoothed_Residuals
## 1          0.2539485
## 2          0.2356578
## 3          0.1630132
## 4          0.1373319
## 5          0.1361468
## 6          0.1185048
```

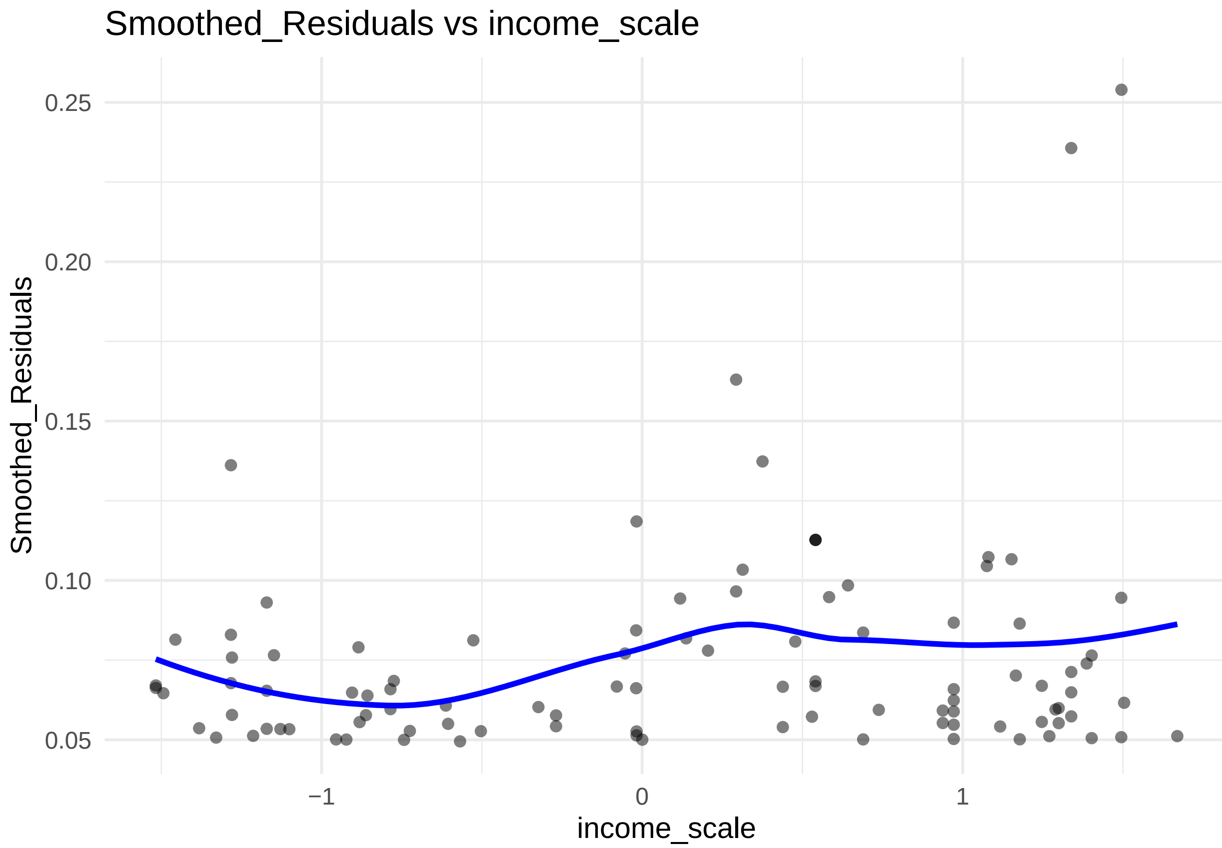
Creating Plotting Function for the Smoothed Residuals

```r
plot_srm_residuals <- function(data, residual_var = "Smoothed_Residuals") {
  predictors <- setdiff(names(data), residual_var)

  for (var in predictors) {
    x <- data[[var]]

    # Determine type
    unique_vals <- unique(na.omit(x))
    is_binary <- length(unique_vals) == 2 && is.numeric(x)
    is_categorical <- is.factor(x) || is.character(x) || is_binary
    is_continuous <- is.numeric(x) && length(unique_vals) > 10

    # Build plot
    if (is_continuous) {
      p <- ggplot(data, aes_string(x = var, y = residual_var)) +
        geom_point(alpha = 0.5) +
        geom_smooth(method = "loess", se = FALSE, color = "blue") +
        labs(title = paste(residual_var, "vs", var),
             x = var, y = residual_var) +
        theme_minimal()

    } else if (is_categorical) {
      # Convert to factor for group-wise plotting if not already
      data[[var]] <- as.factor(data[[var]])

      p <- ggplot(data, aes_string(x = var, y = residual_var)) +
        geom_boxplot(fill = "darkorange", alpha = 0.7) +
        labs(title = paste(residual_var, "by", var),
             x = var, y = residual_var) +
        theme_minimal()
    } else {
      next  # Skip variables that don't meet either condition
    }

    print(p)
  }
}
```

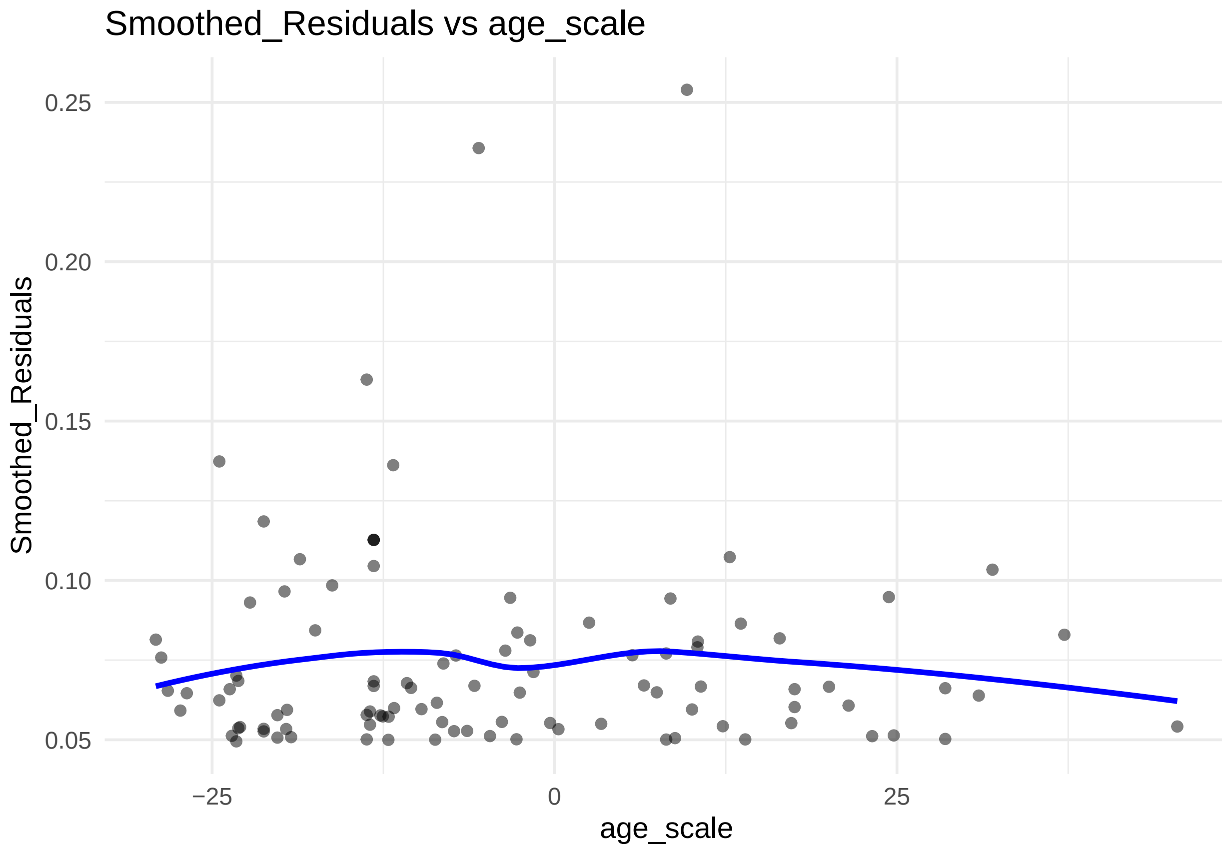
```r
plot_srm_residuals(srm_final)
```



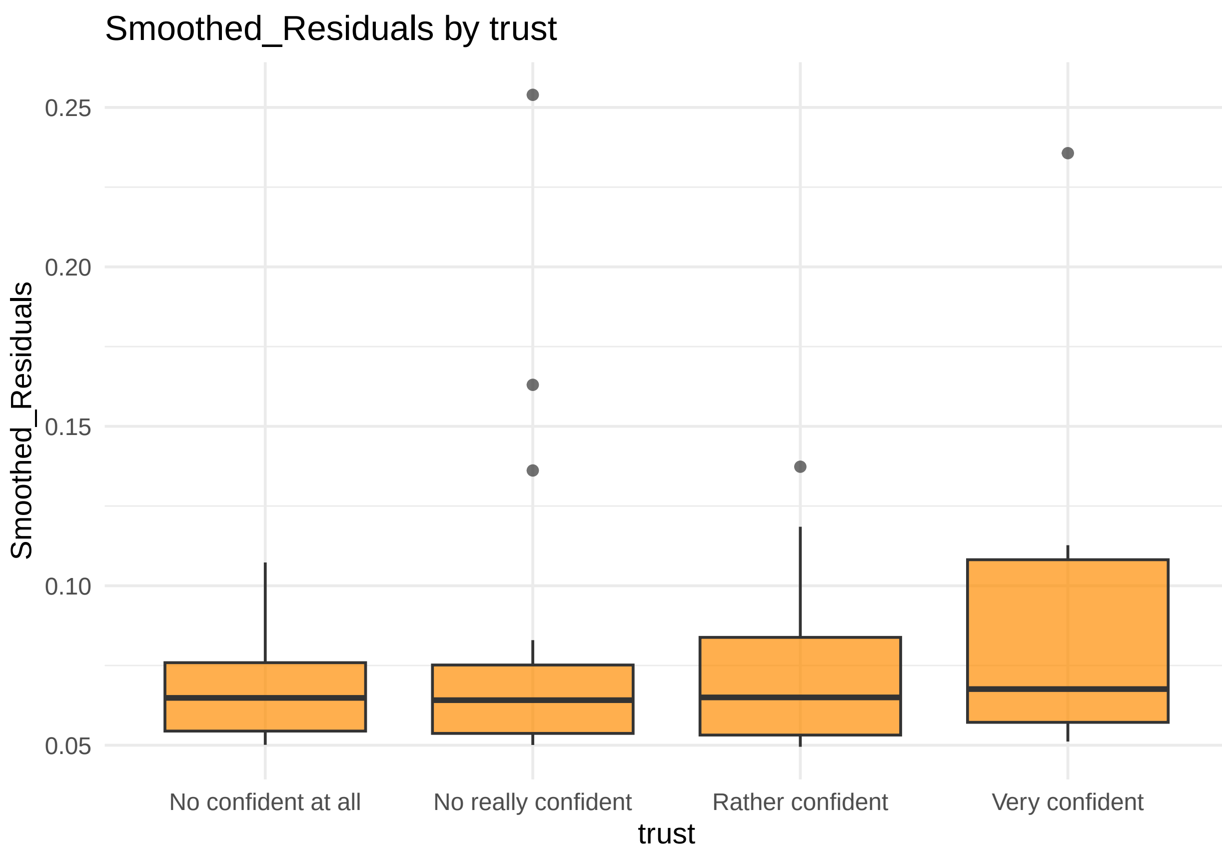

```
## `geom_smooth()` using formula = 'y ~ x'
```



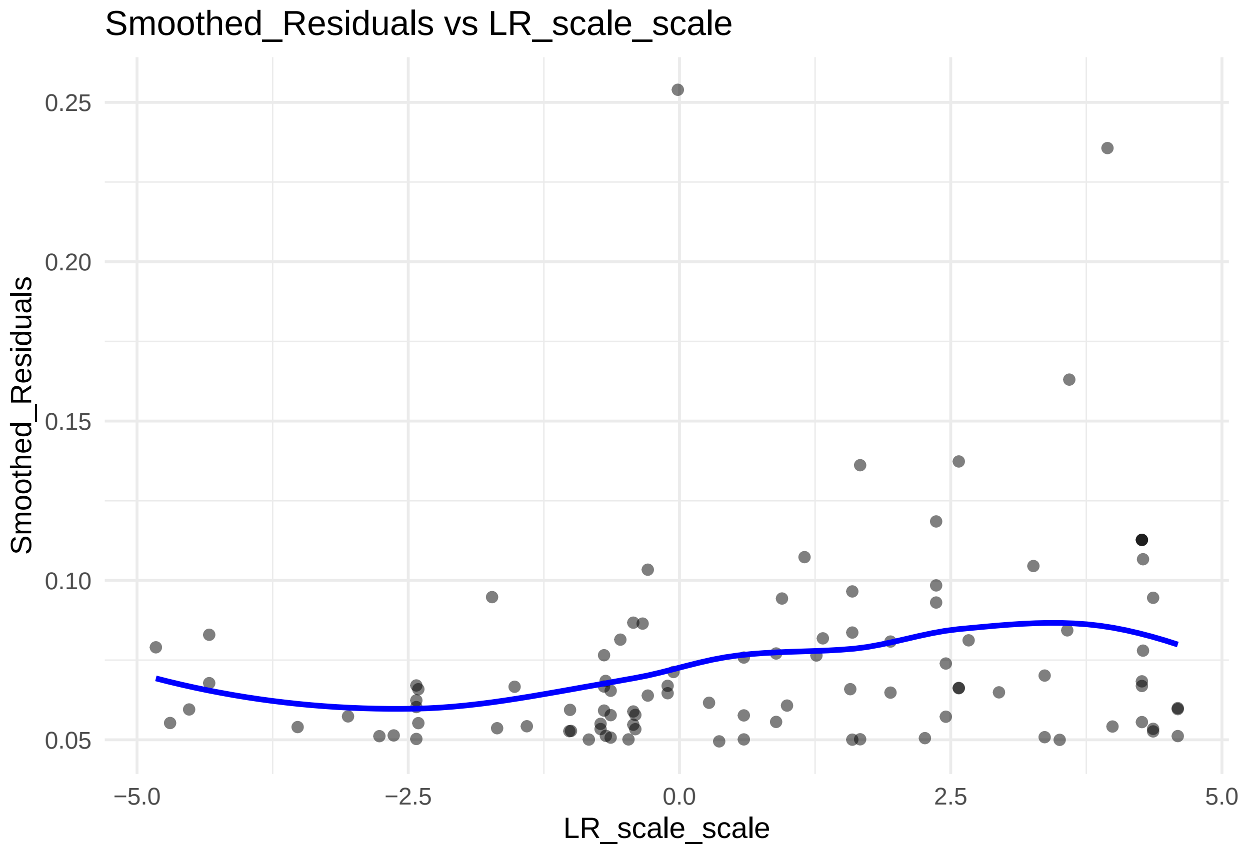

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



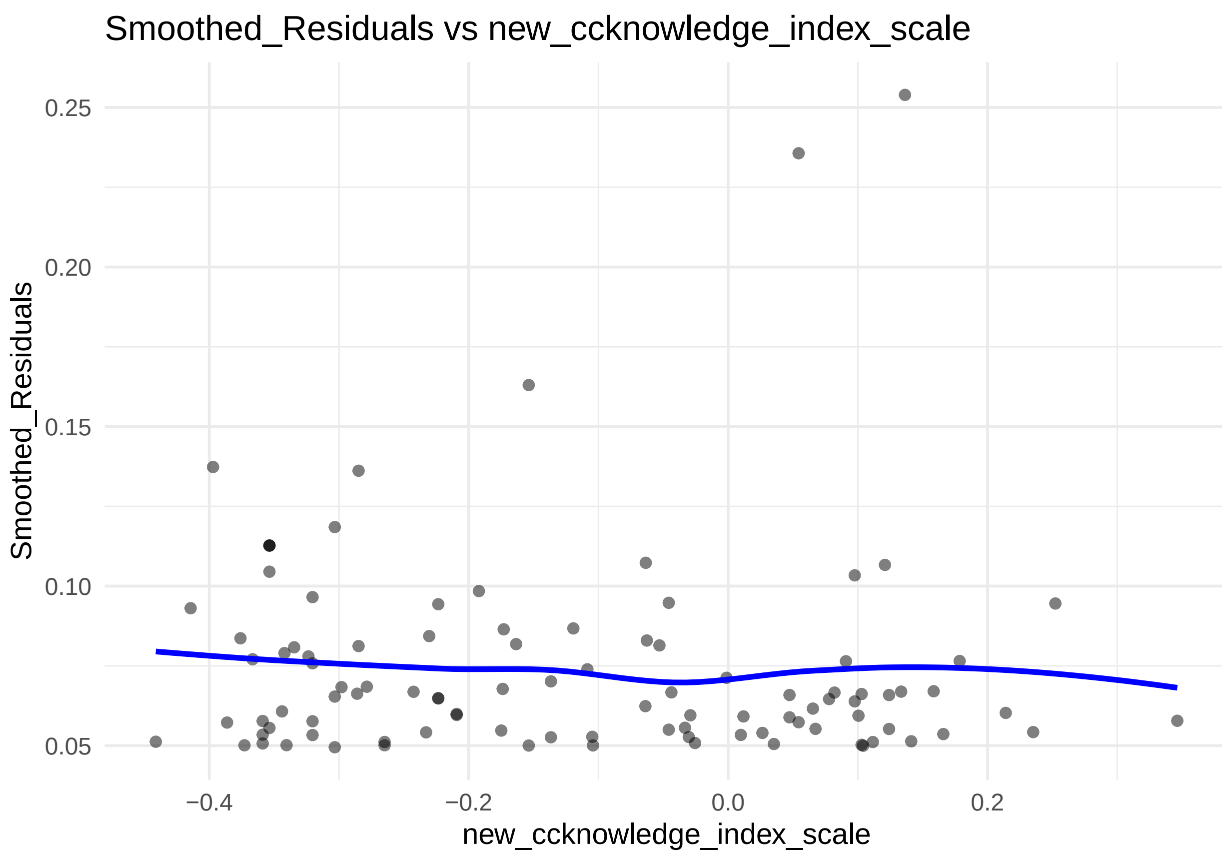
Smoothed_Residuals vs age_scale
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
−25
0
25
age_scale

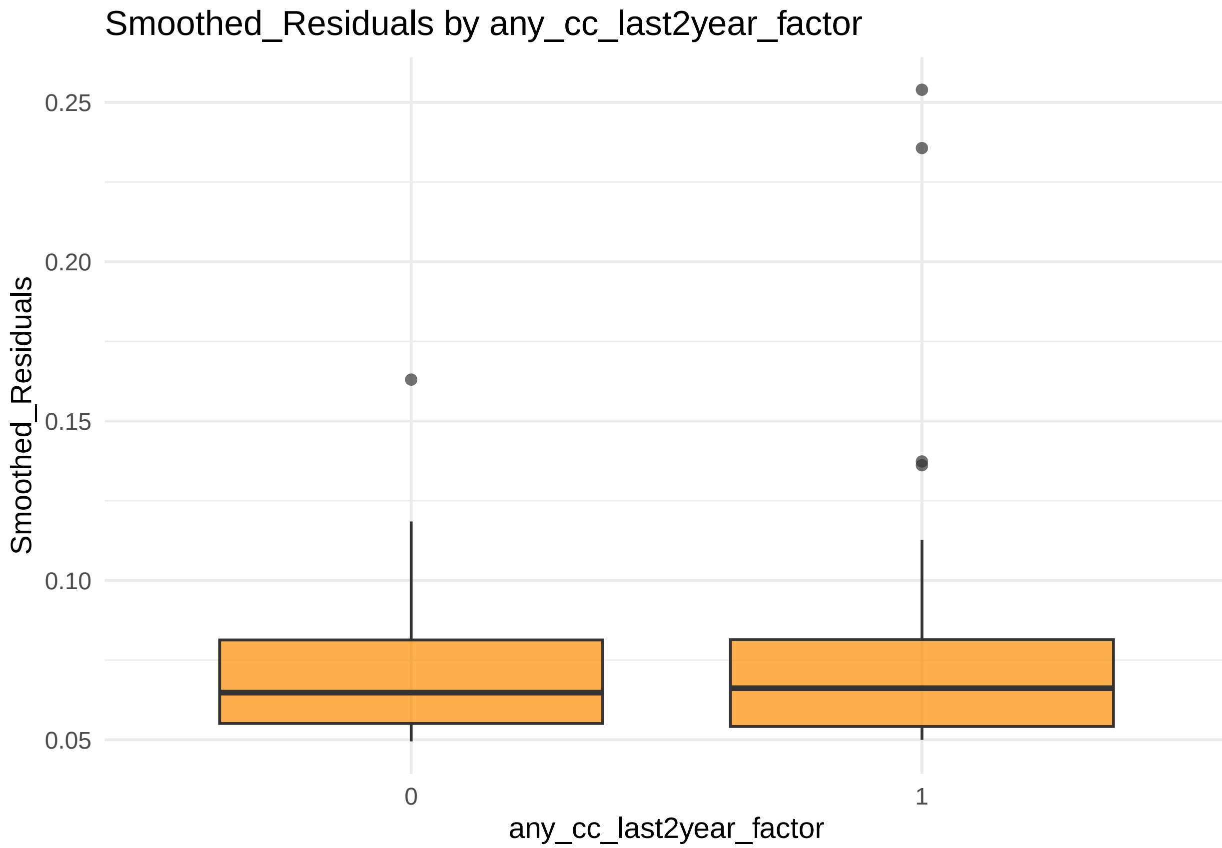



```
## `geom_smooth()` using formula = 'y ~ x'
```

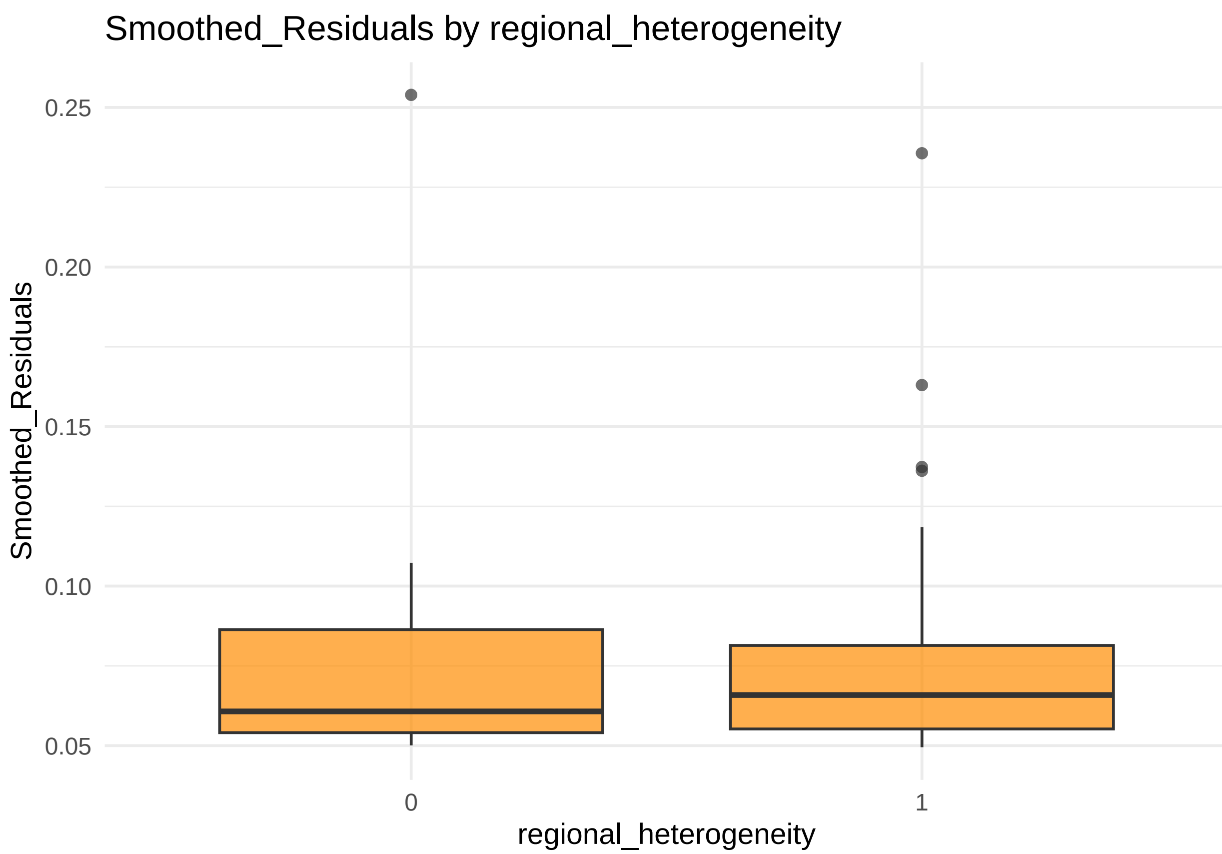



```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```

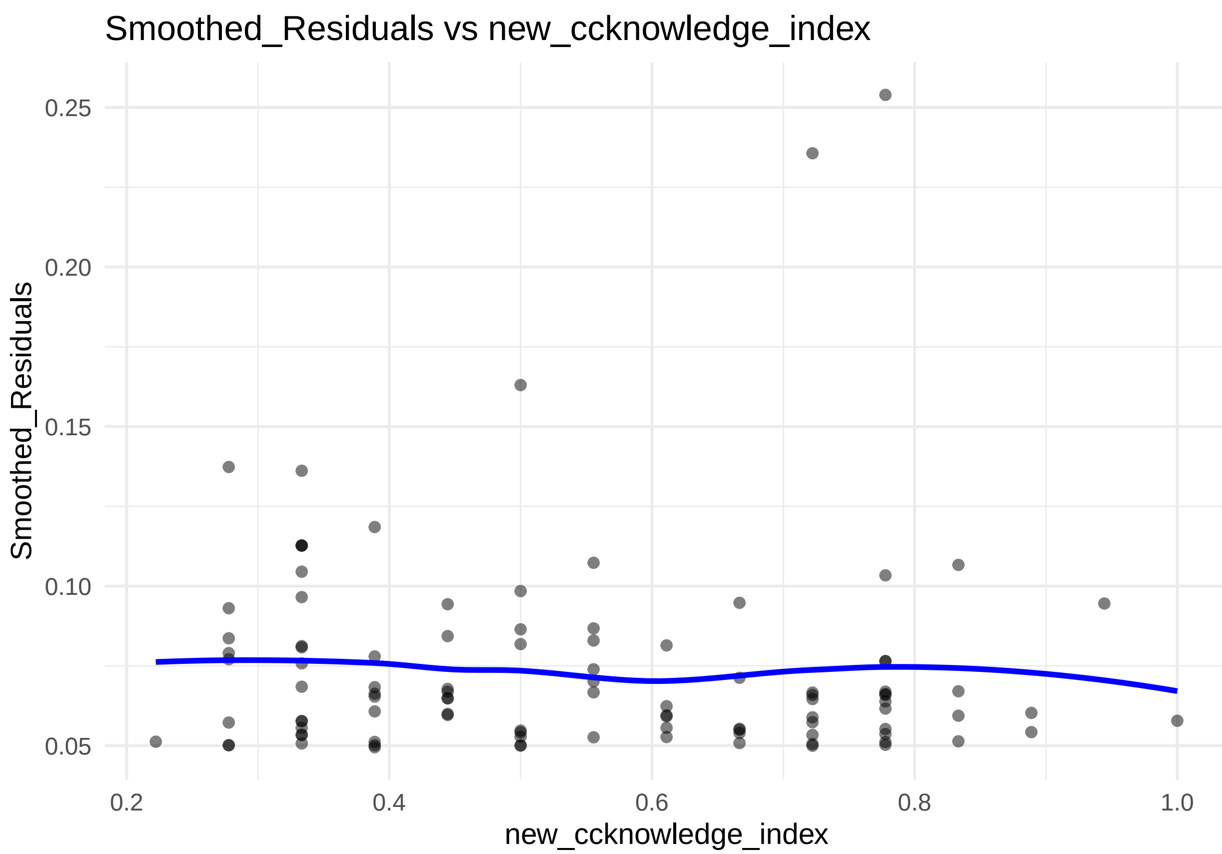

Smoothed_Residuals vs new_ccknowledge_index_scale
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
−0.4
−0.2
0.0
0.2
new_ccknowledge_index_scale

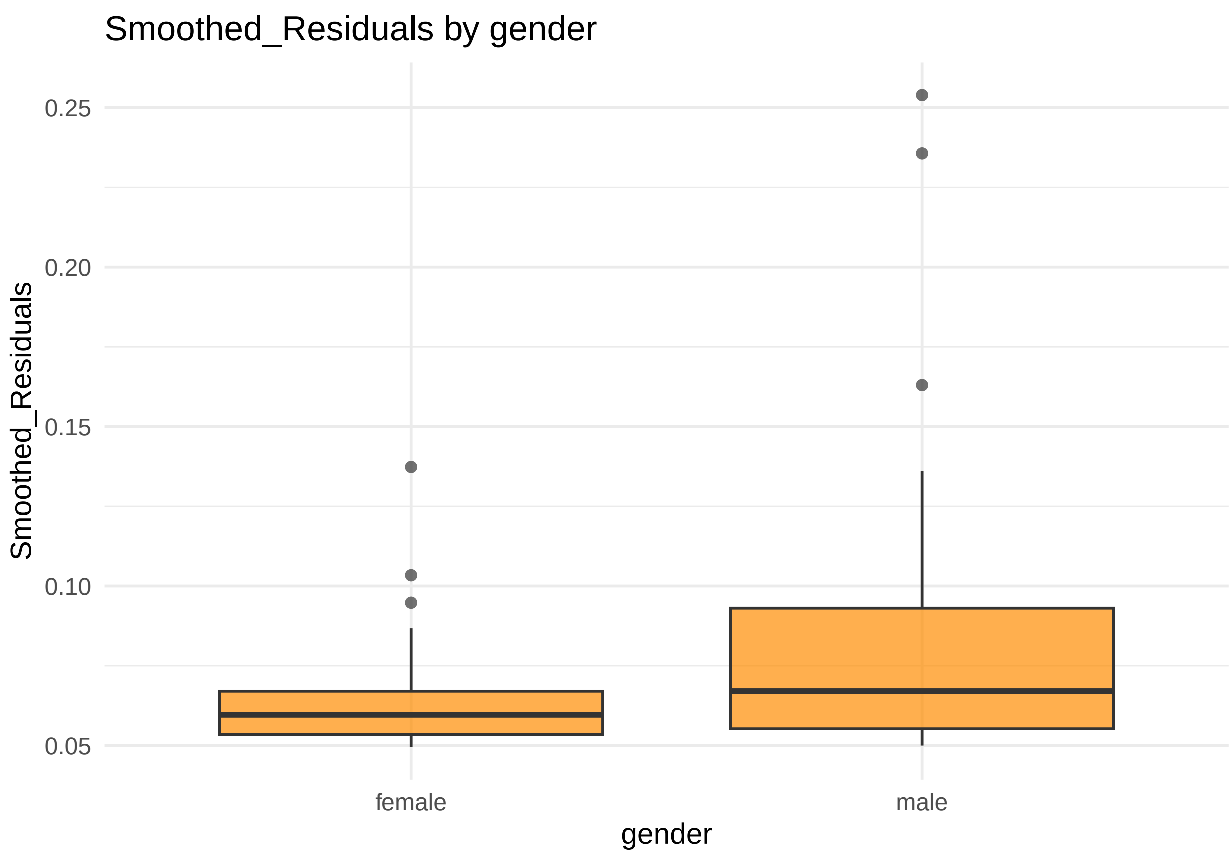


Smoothed_Residuals by any_cc_last2year_factor
0.25
0.20
0.15
0.10
0.05
Smoothed_Residuals
0
1
any_cc_last2year_factor

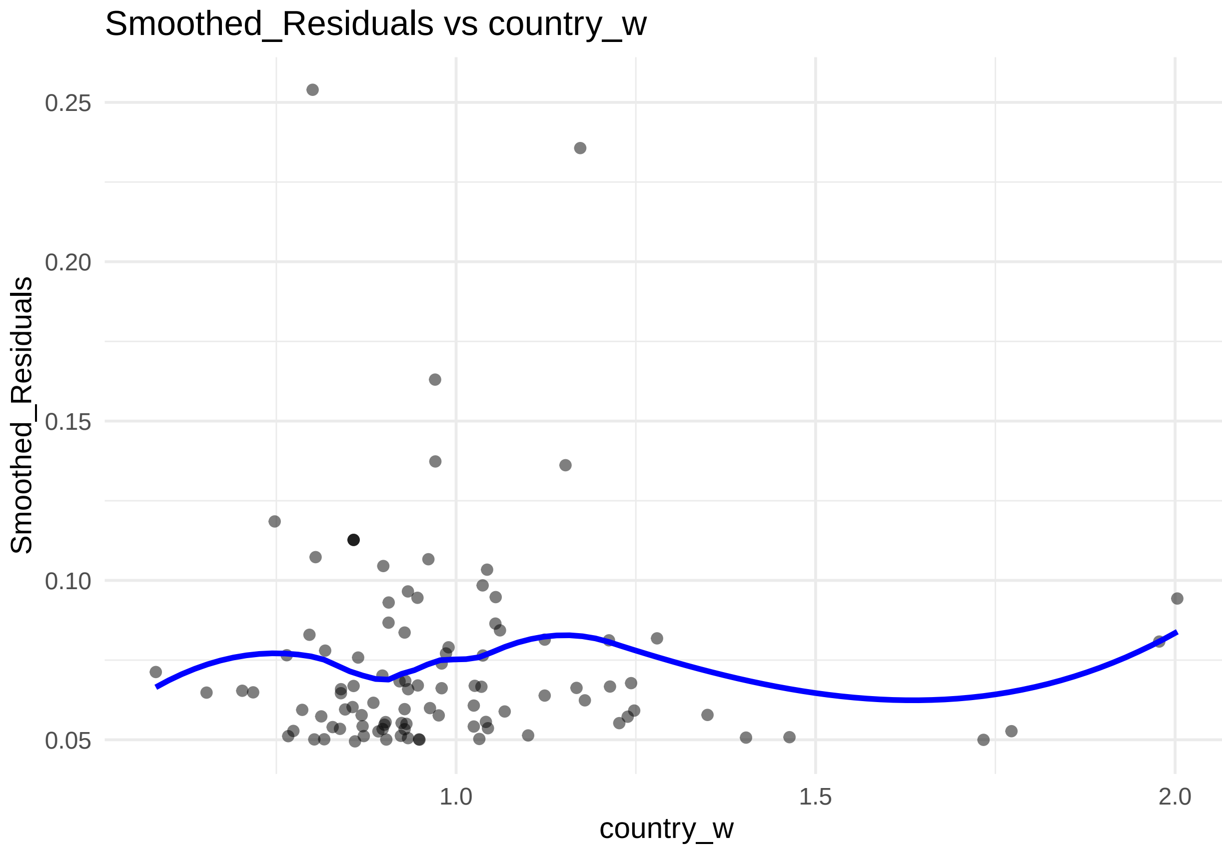


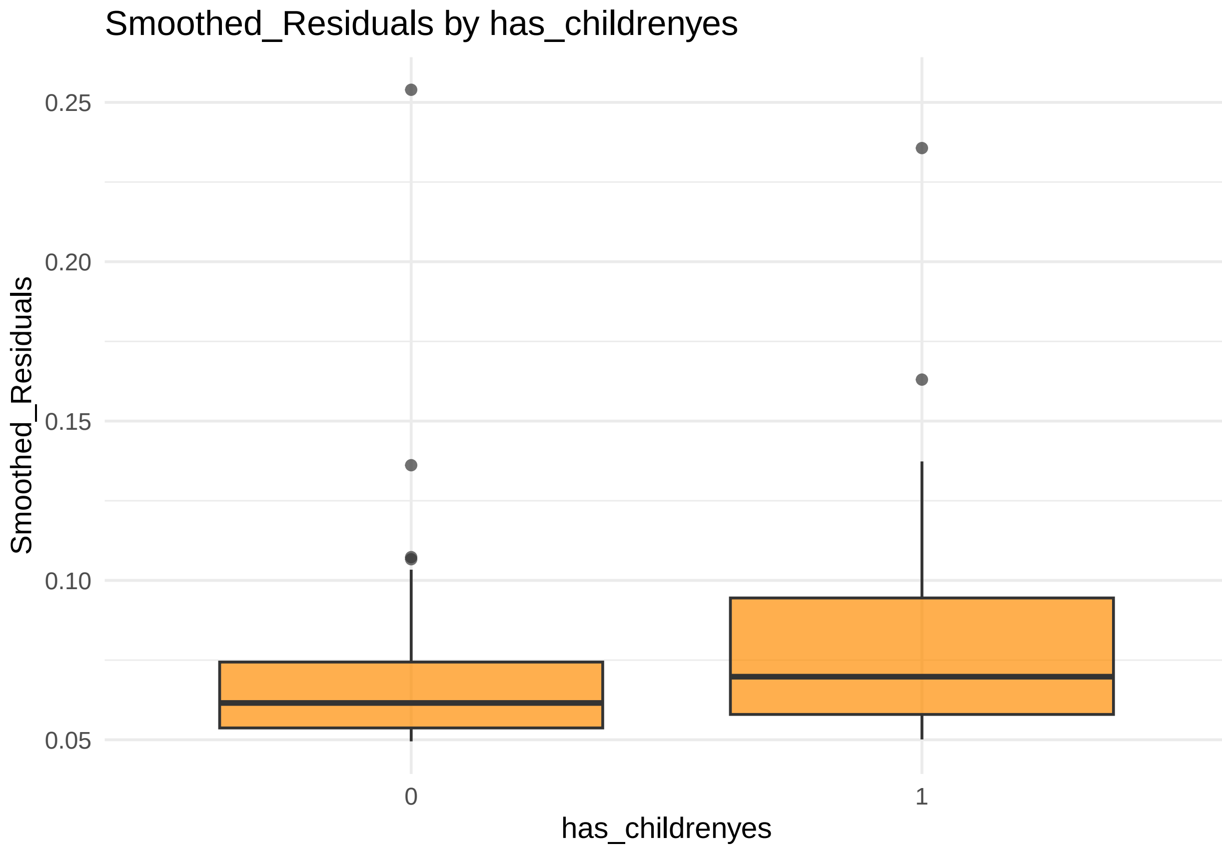

```
## `geom_smooth()` using formula = 'y ~ x'
```

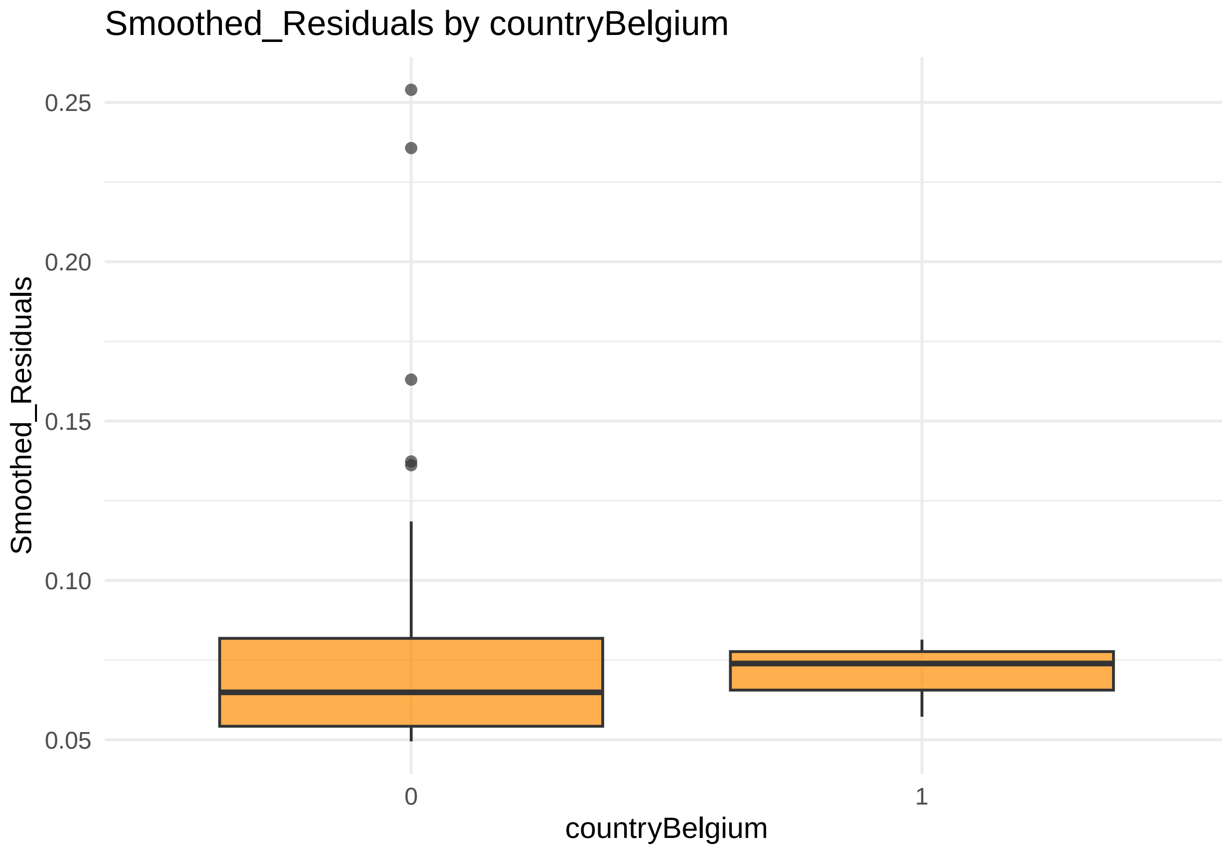

Smoothed_Residuals vs new_ccknowledge_index
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0.2
0.4
0.6
0.8
1.0
new_ccknowledge_index

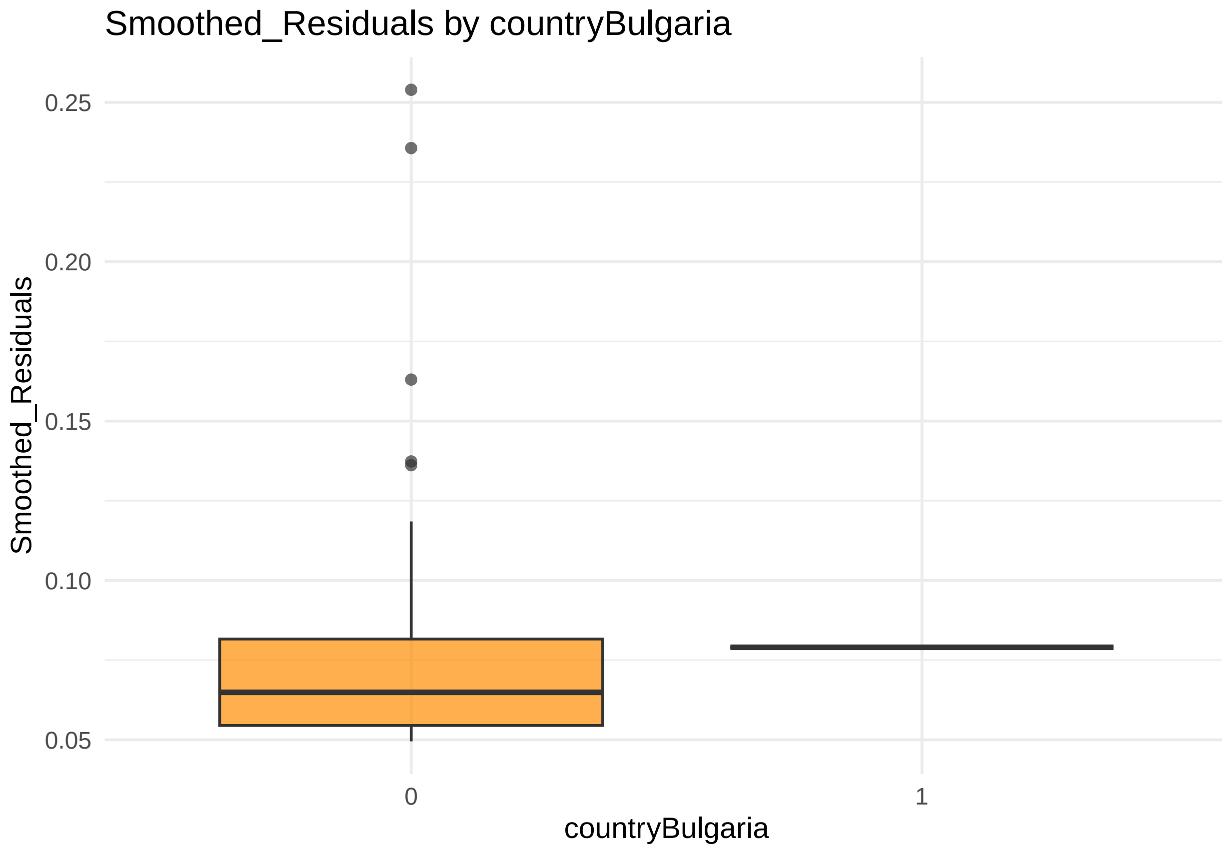


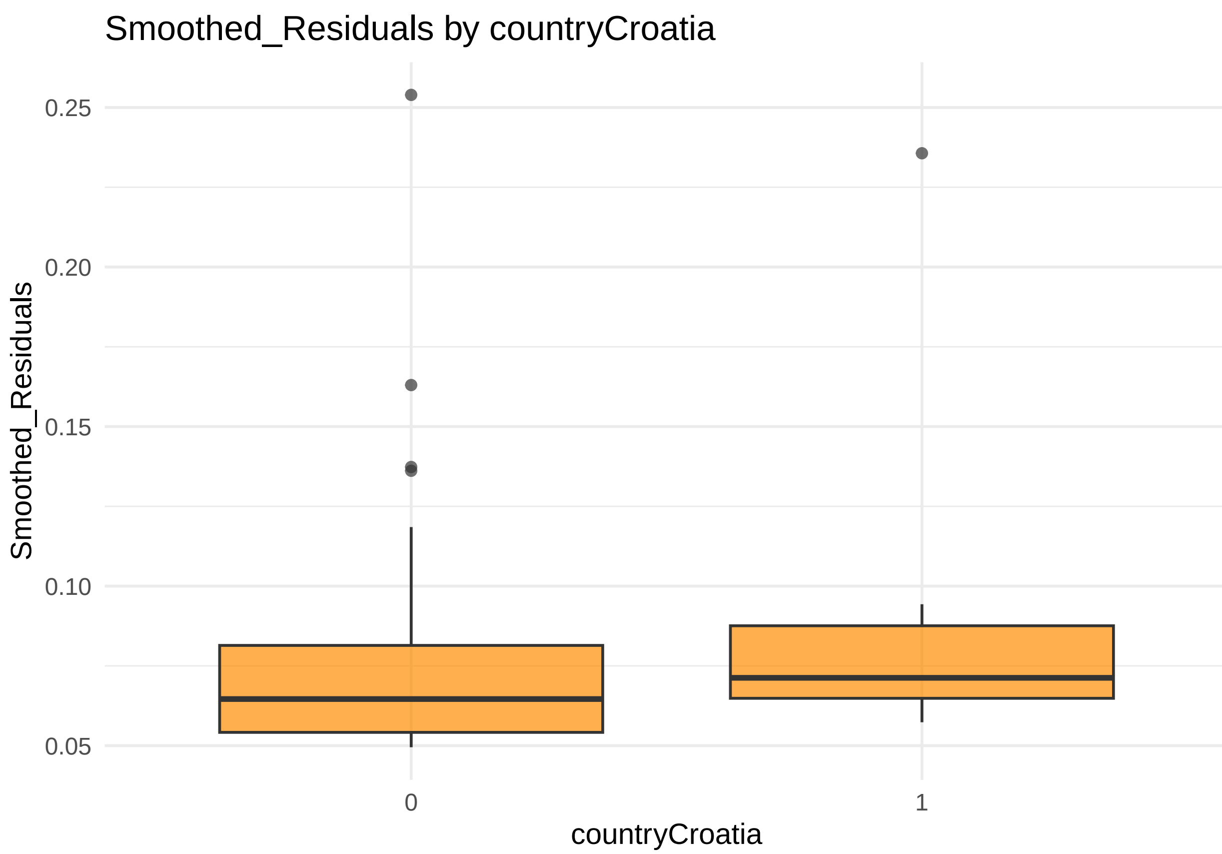

```
## `geom_smooth()` using formula = 'y ~ x'
```

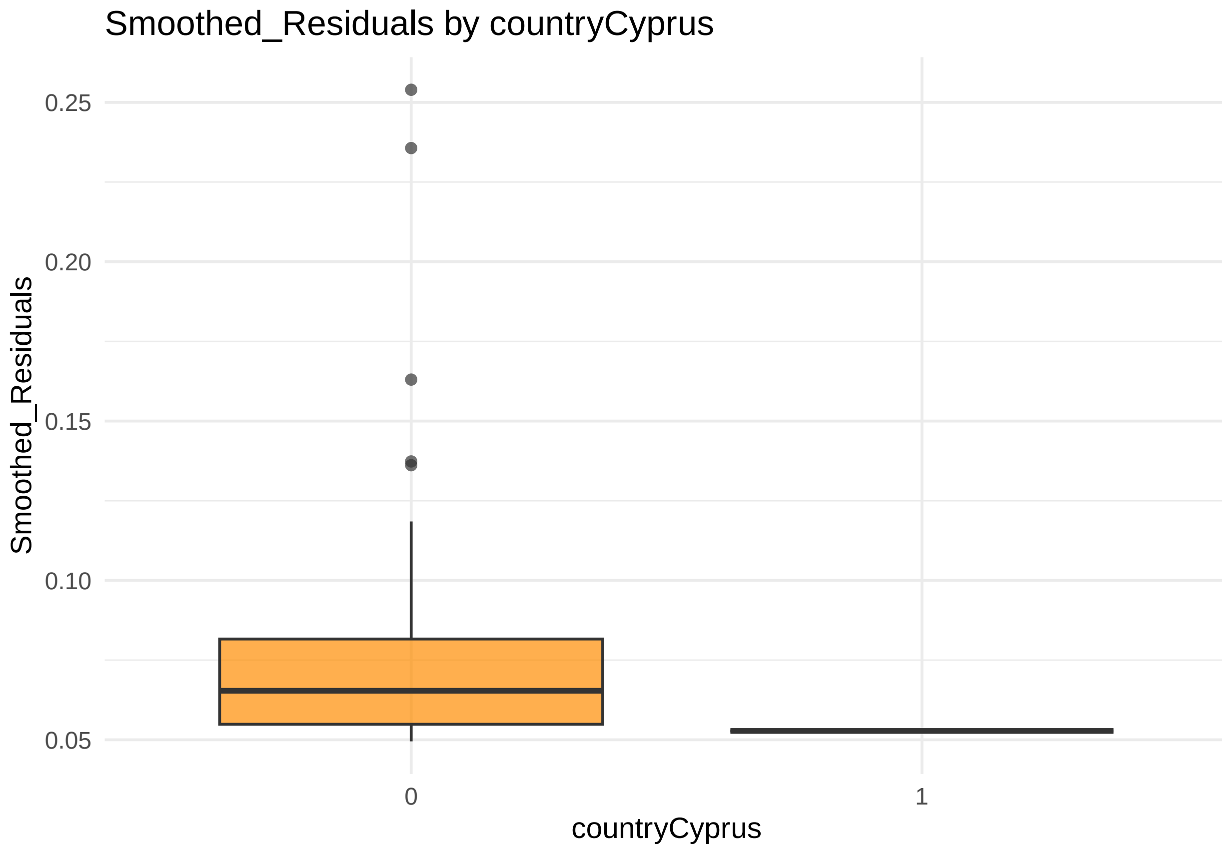

Smoothed_Residuals vs country_w
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
1.0
1.5
2.0
country_w

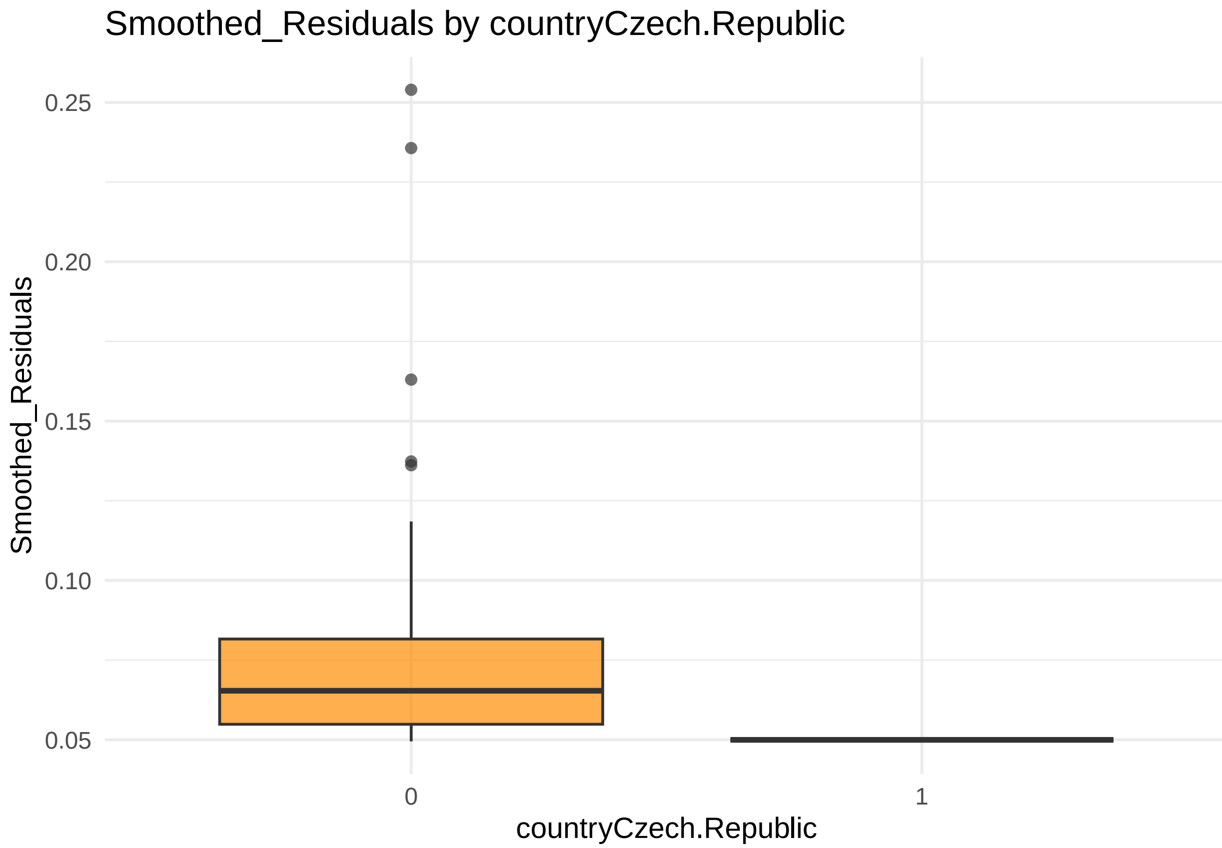

Smoothed_Residuals by has_childrenyes
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
has_childrenyes

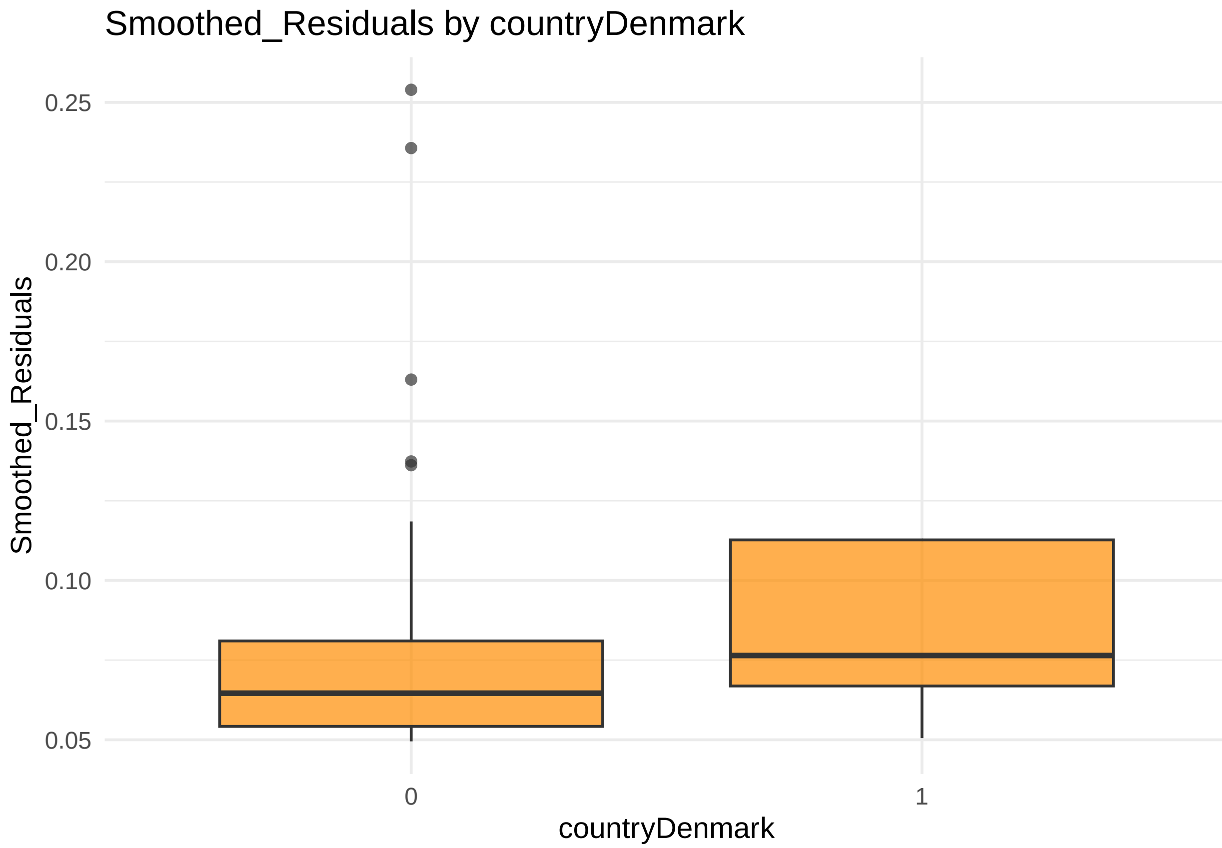

Smoothed_Residuals by countryBelgium
0.25
0.20
0.15
0.10
0.05
Smoothed_Residuals
0
1
countryBelgium

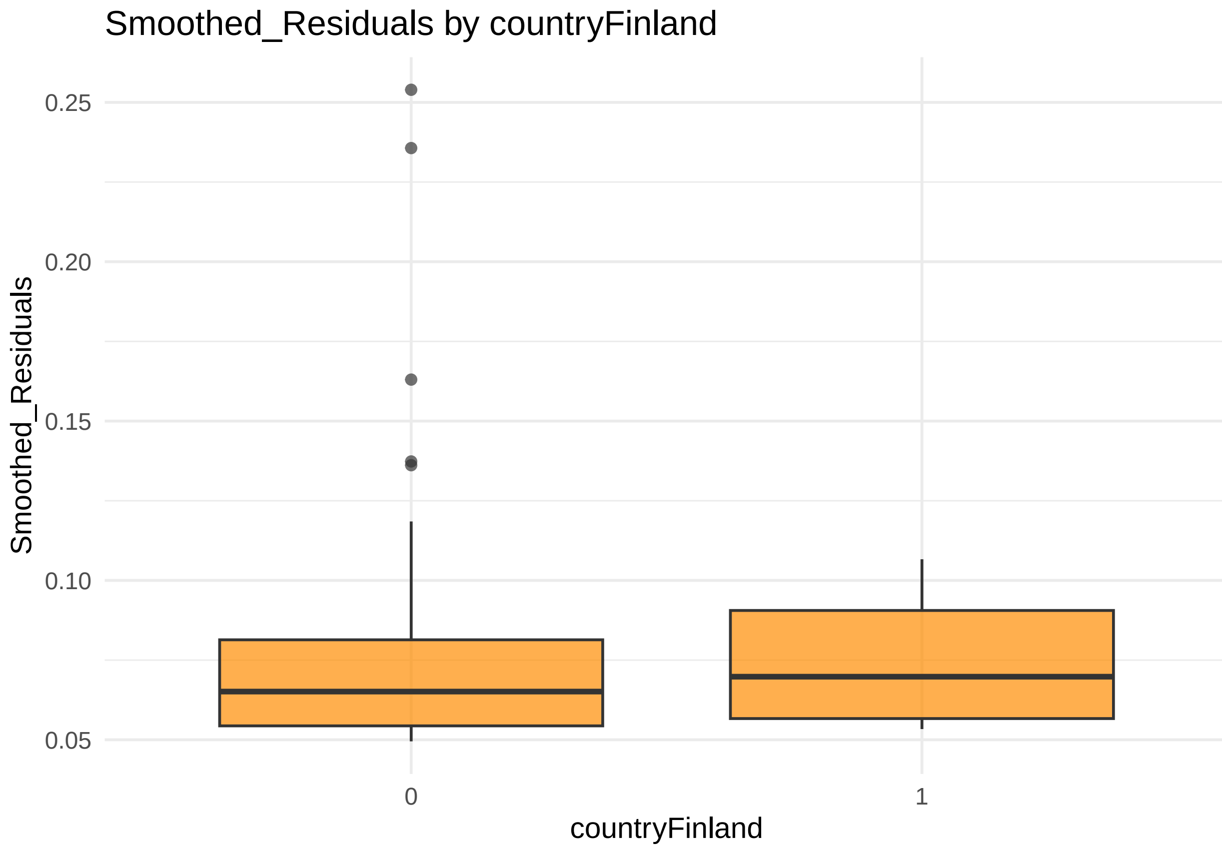

Smoothed_Residuals by countryBulgaria
0.25
0.20
0.15
0.10
0.05
Smoothed_Residuals
0
1
countryBulgaria

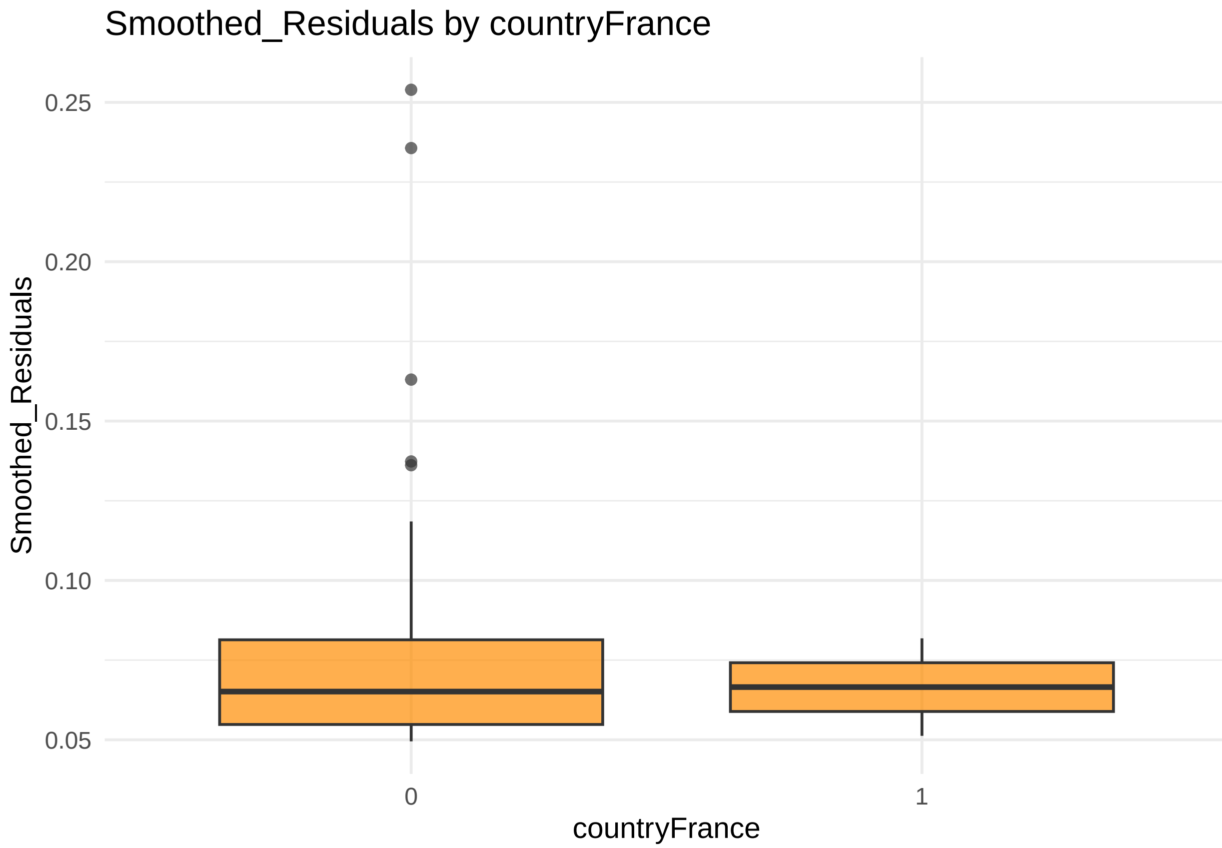

Smoothed_Residuals by countryCroatia
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryCroatia

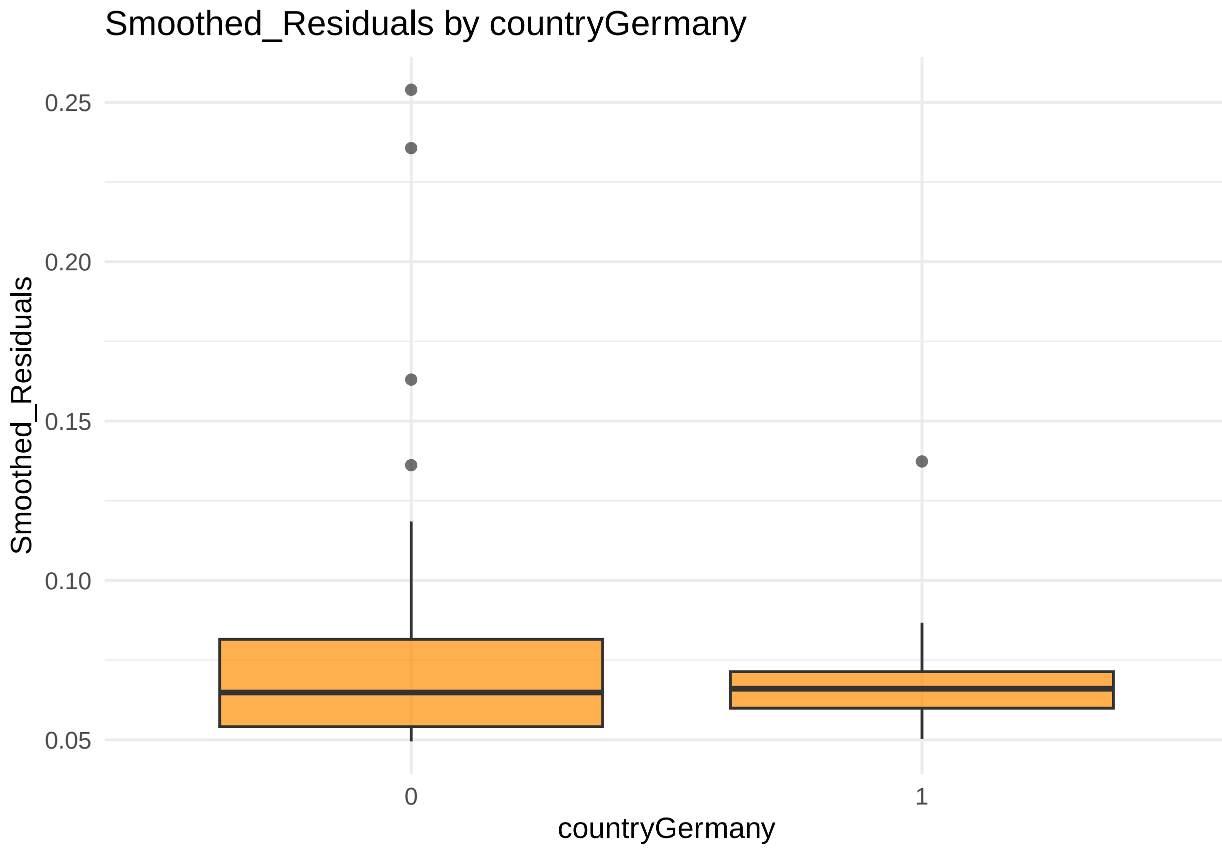

Smoothed_Residuals by countryCyprus
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryCyprus

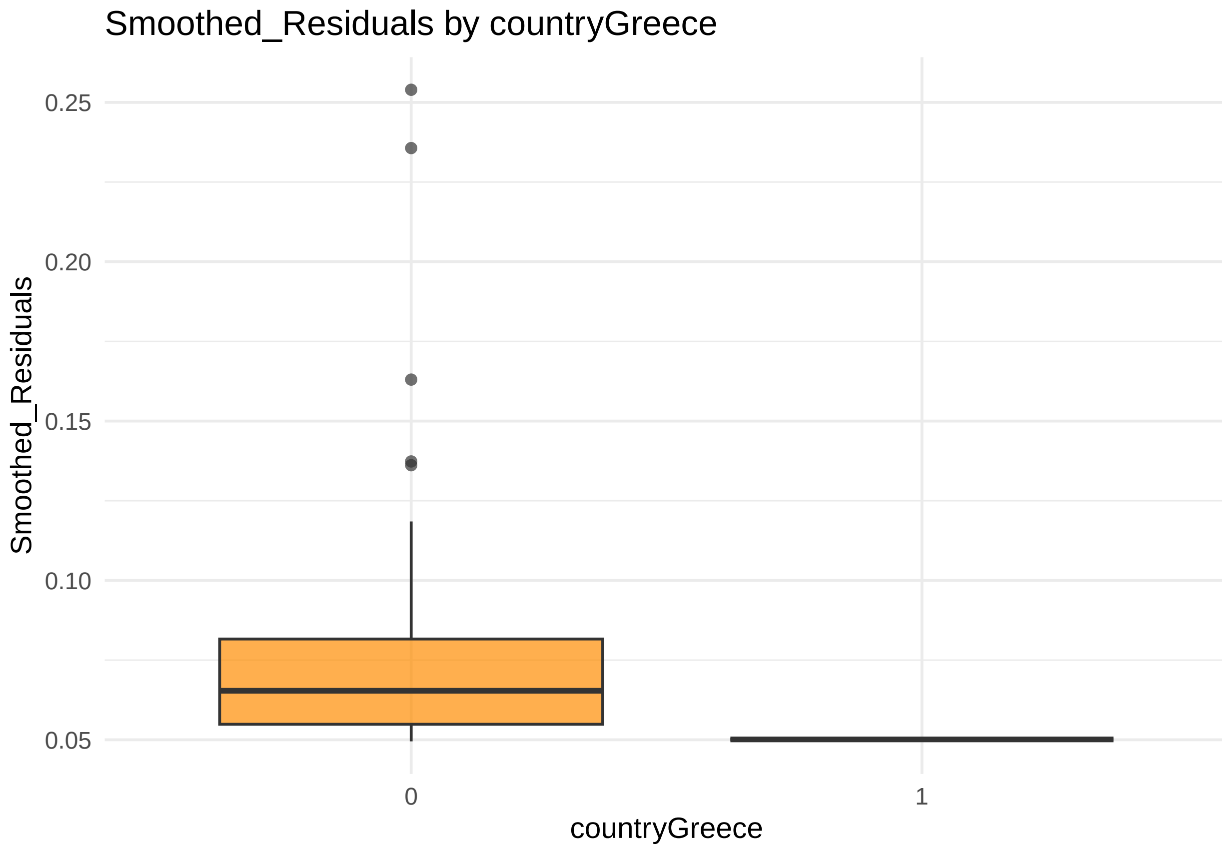

Smoothed_Residuals by countryCzech.Republic
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryCzech.Republic

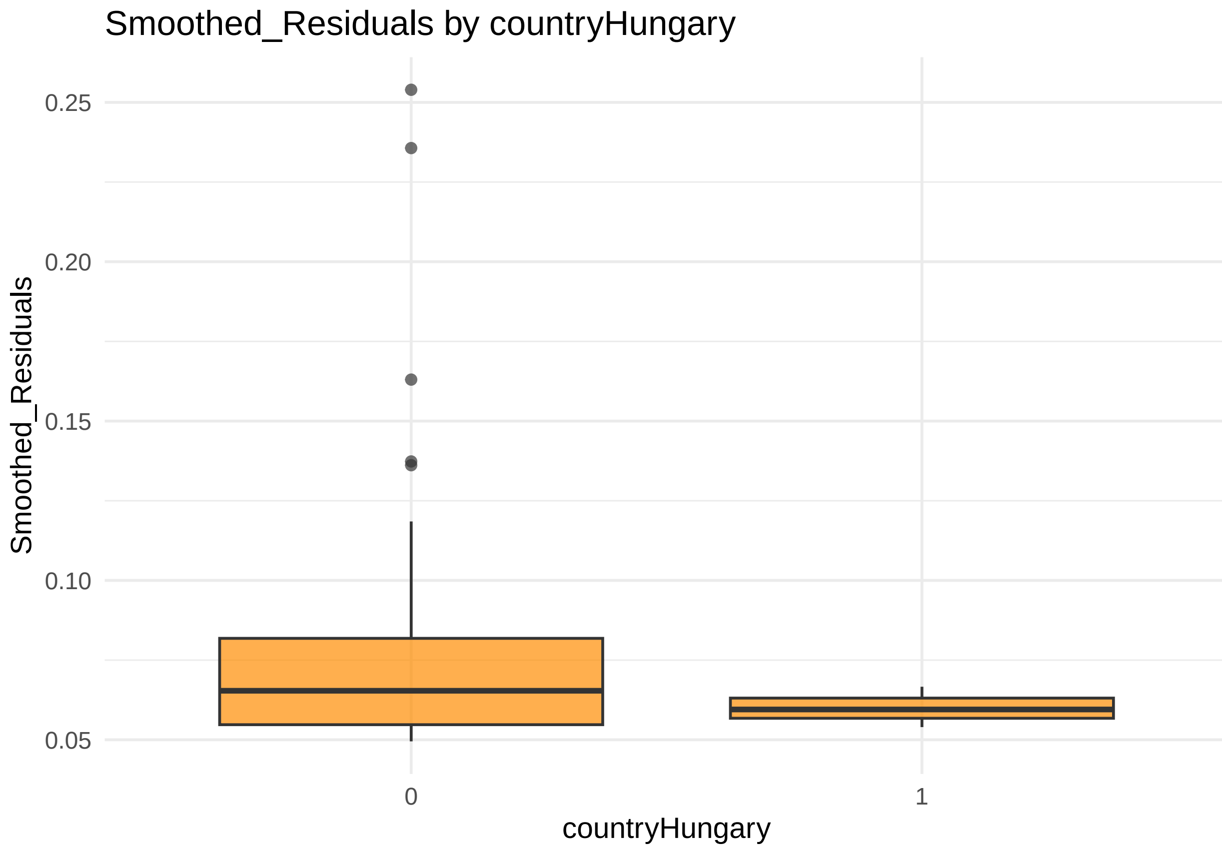

Smoothed_Residuals by countryDenmark
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryDenmark

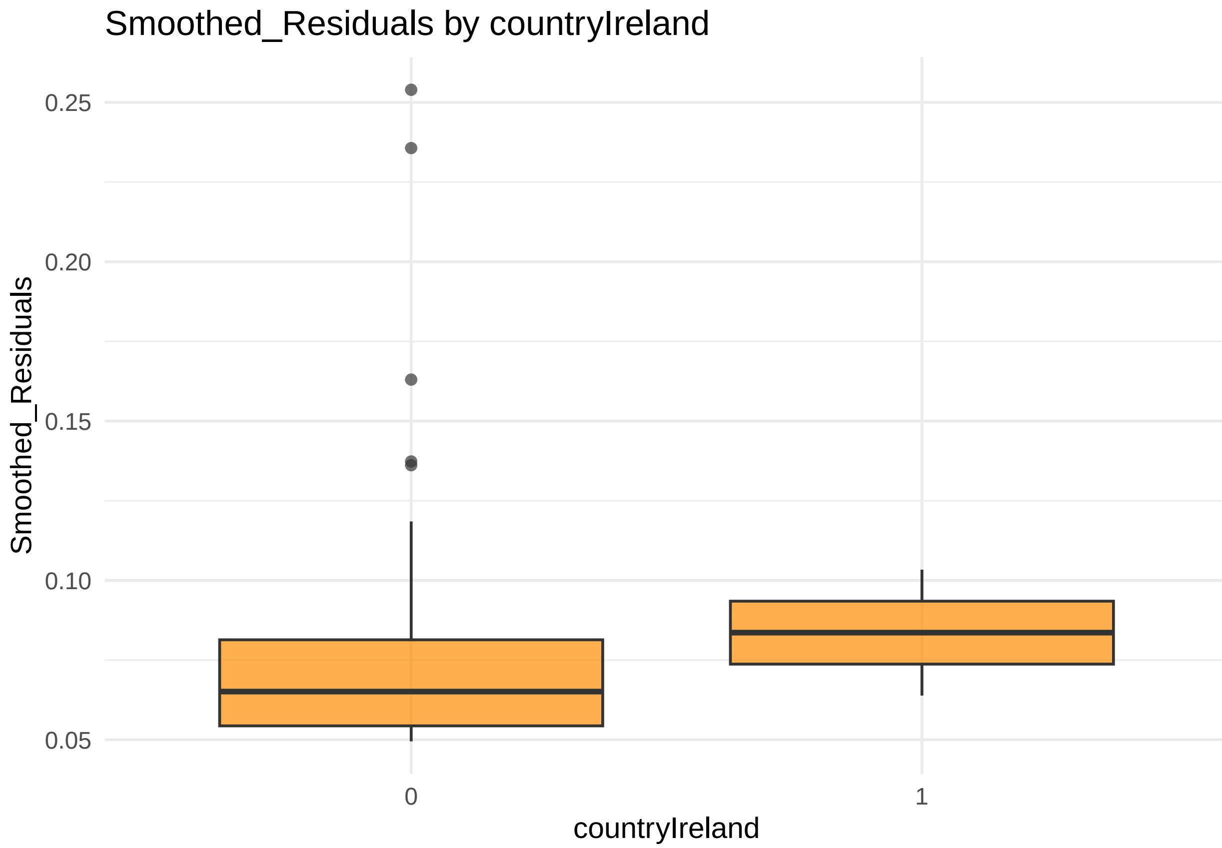

Smoothed_Residuals by countryFinland
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryFinland

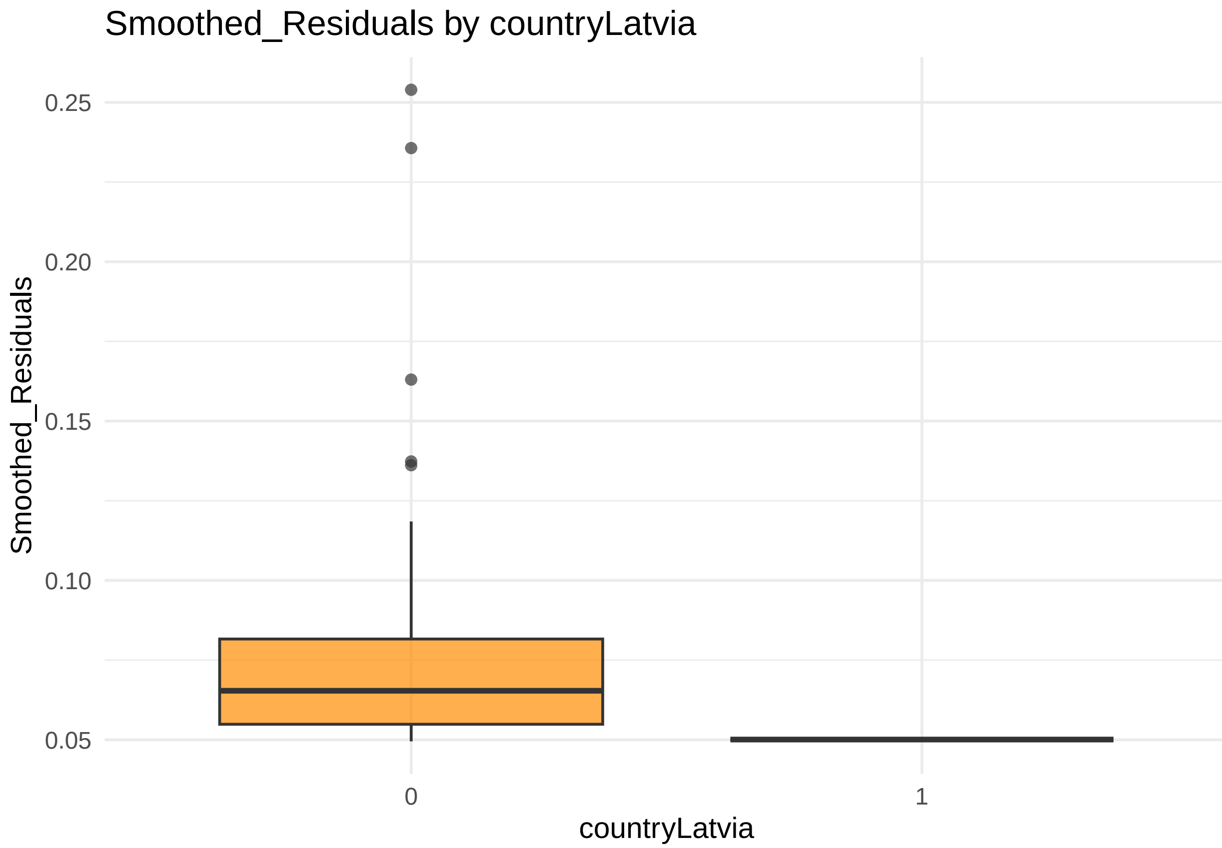

Smoothed_Residuals by countryFrance
0.25
0.20
0.15
0.10
0.05
Smoothed_Residuals
0
1
countryFrance

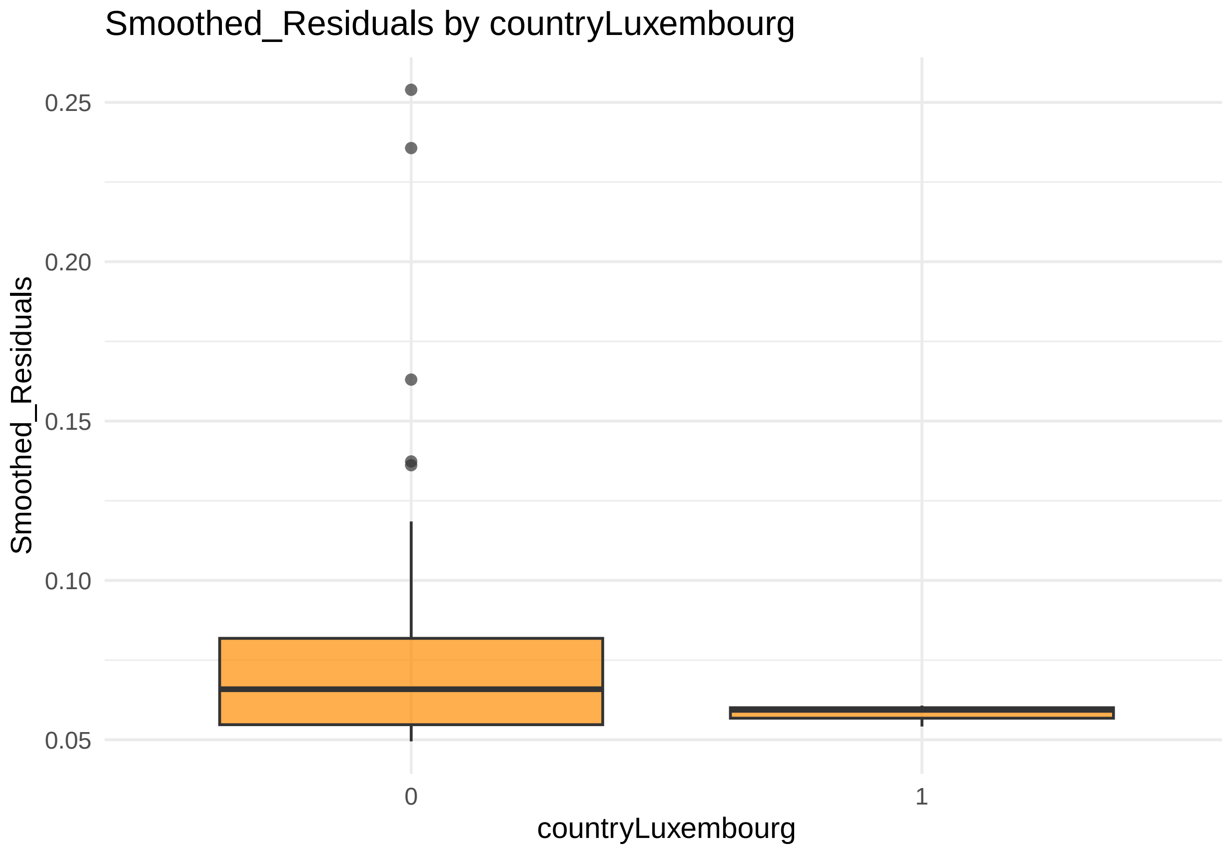

Smoothed_Residuals by countryGermany
0.25
0.20
0.15
0.10
0.05
Smoothed_Residuals
0
1
countryGermany

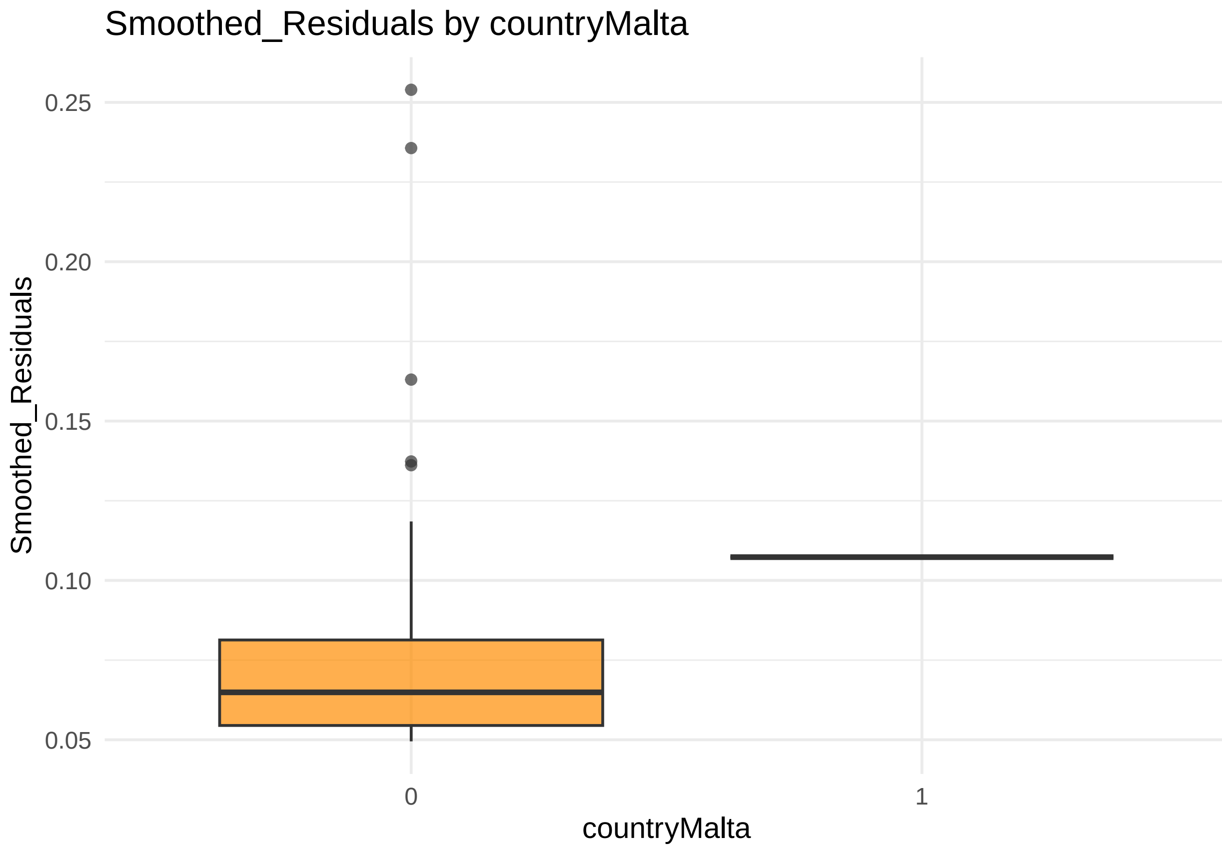

Smoothed_Residuals by countryGreece
0.25
0.20
0.15
0.10
0.05
Smoothed_Residuals
0
1
countryGreece

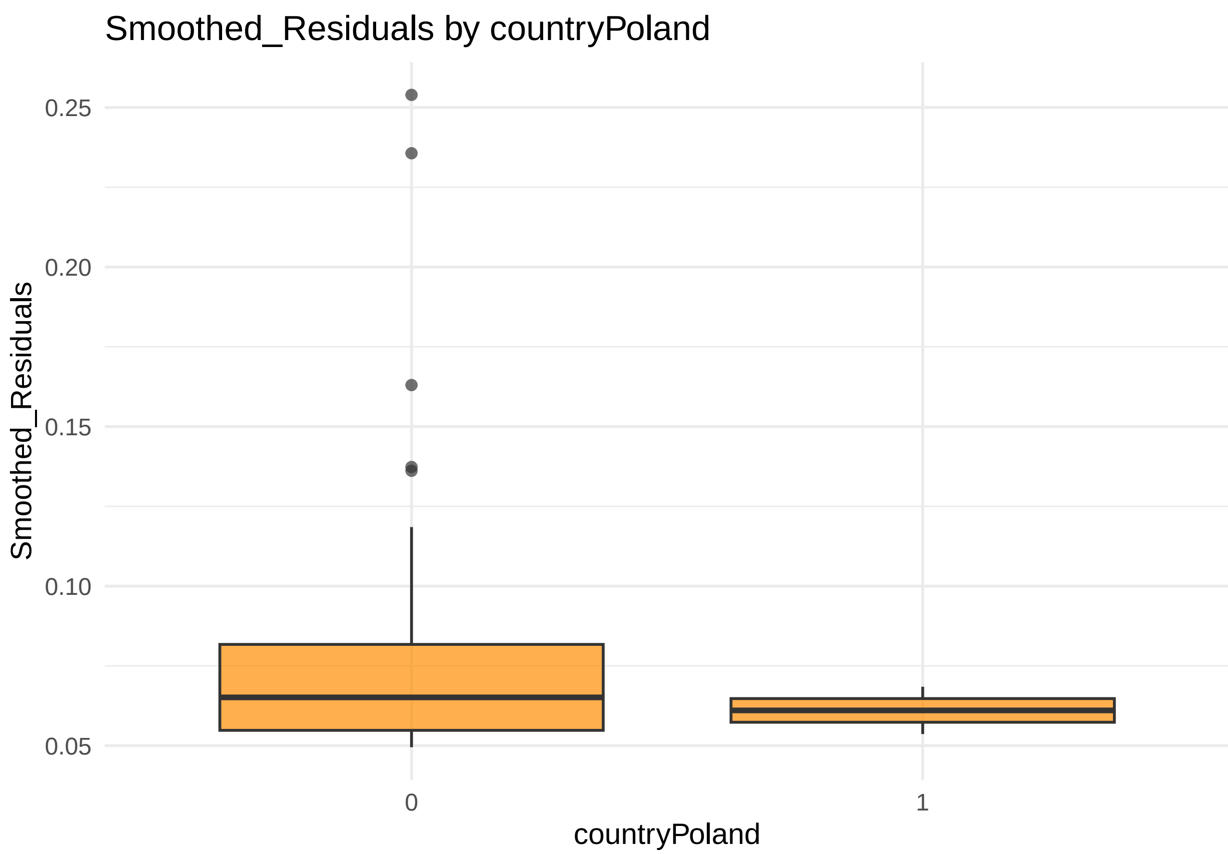

Smoothed_Residuals by countryHungary
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryHungary

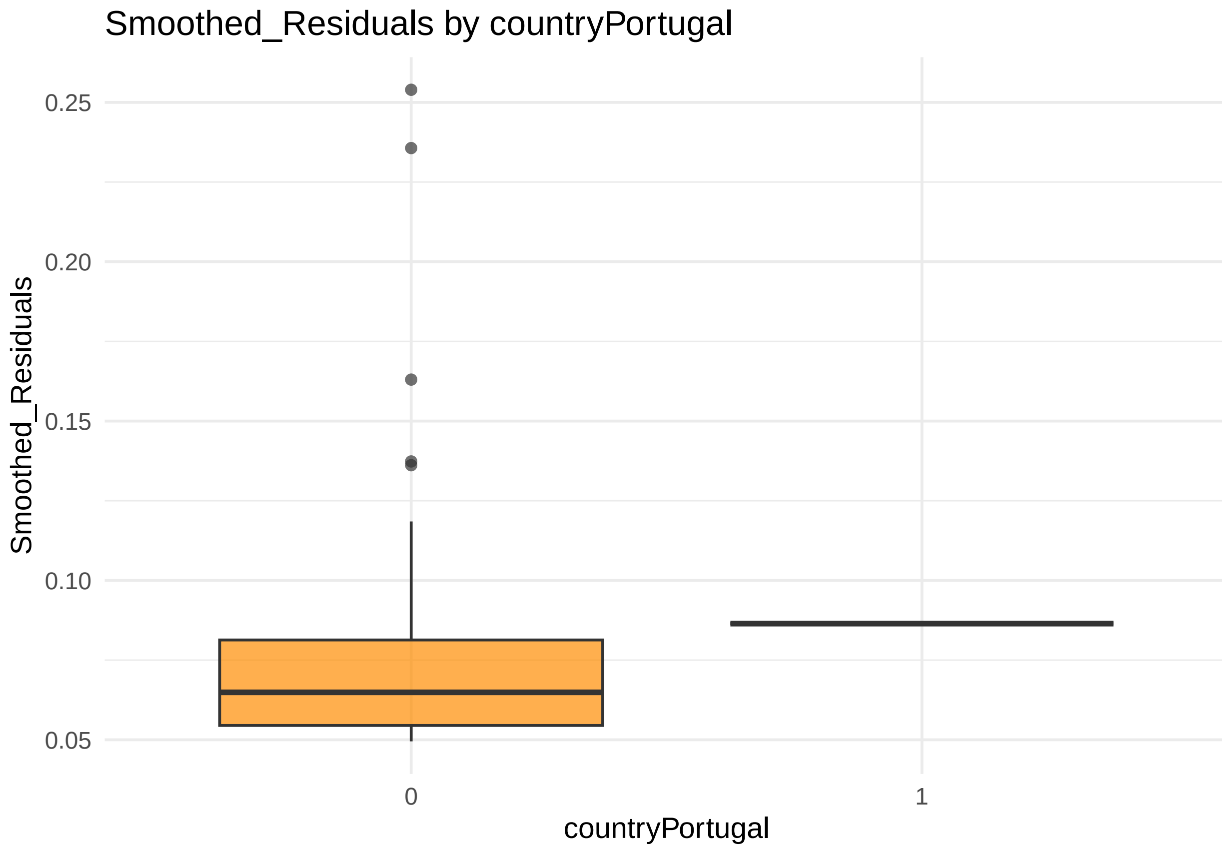

Smoothed_Residuals by countryIreland
0.25
0.20
0.15
0.10
0.05
Smoothed_Residuals
0
1
countryIreland

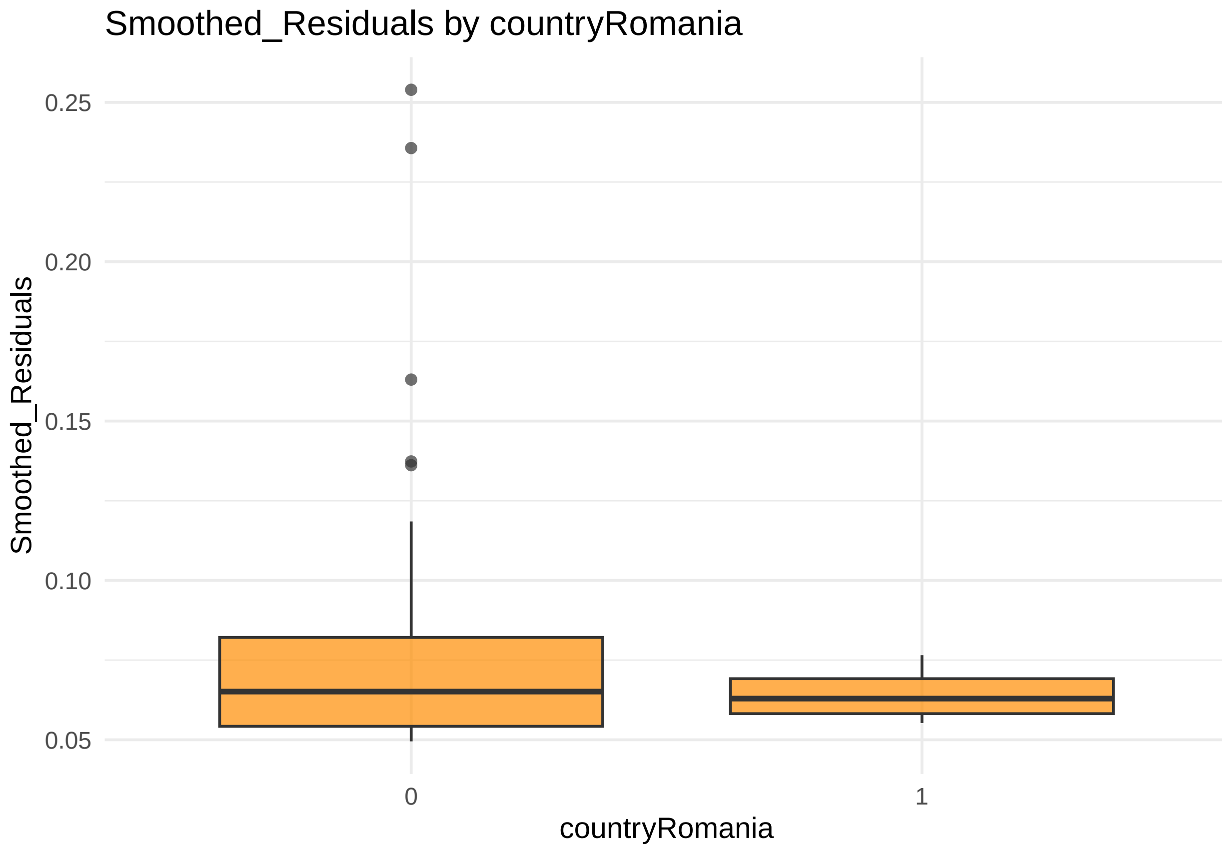

Smoothed_Residuals by countryLatvia
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryLatvia

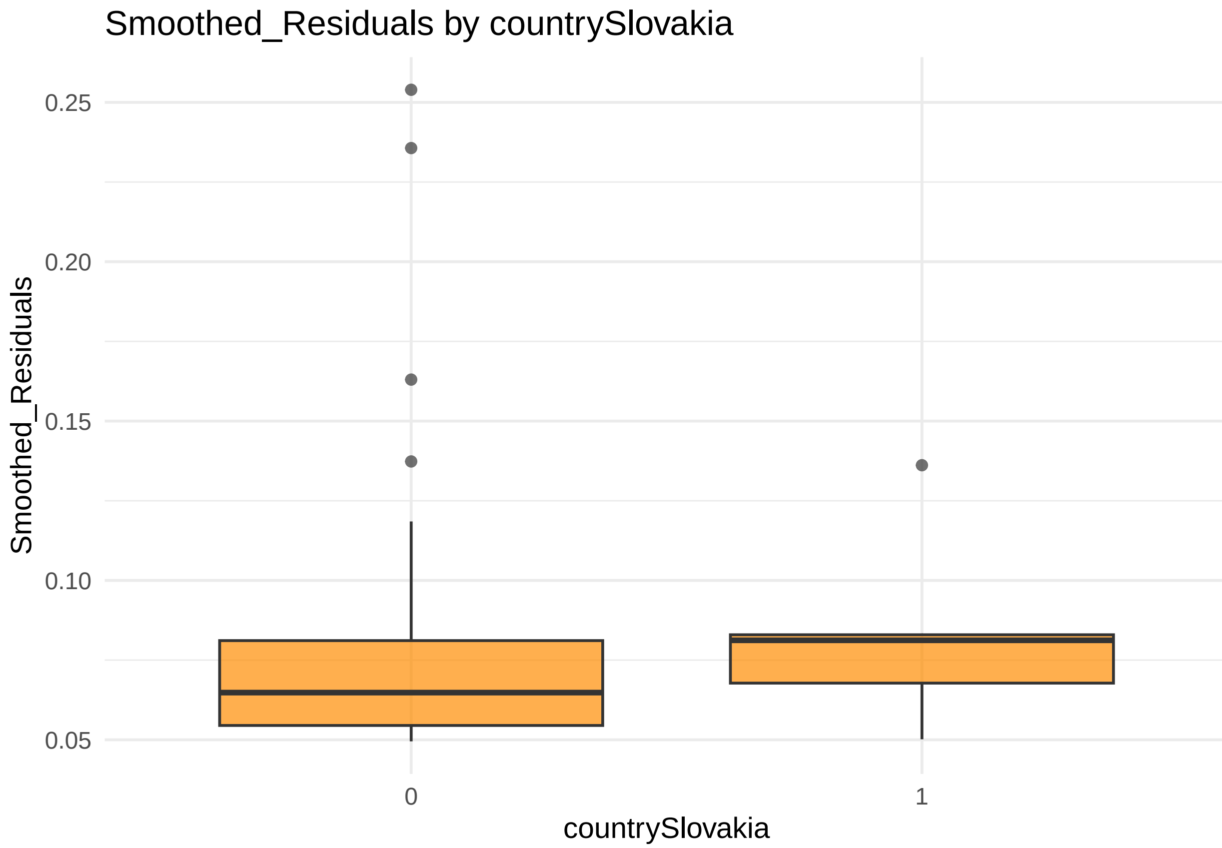

Smoothed_Residuals by countryLuxembourg
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryLuxembourg

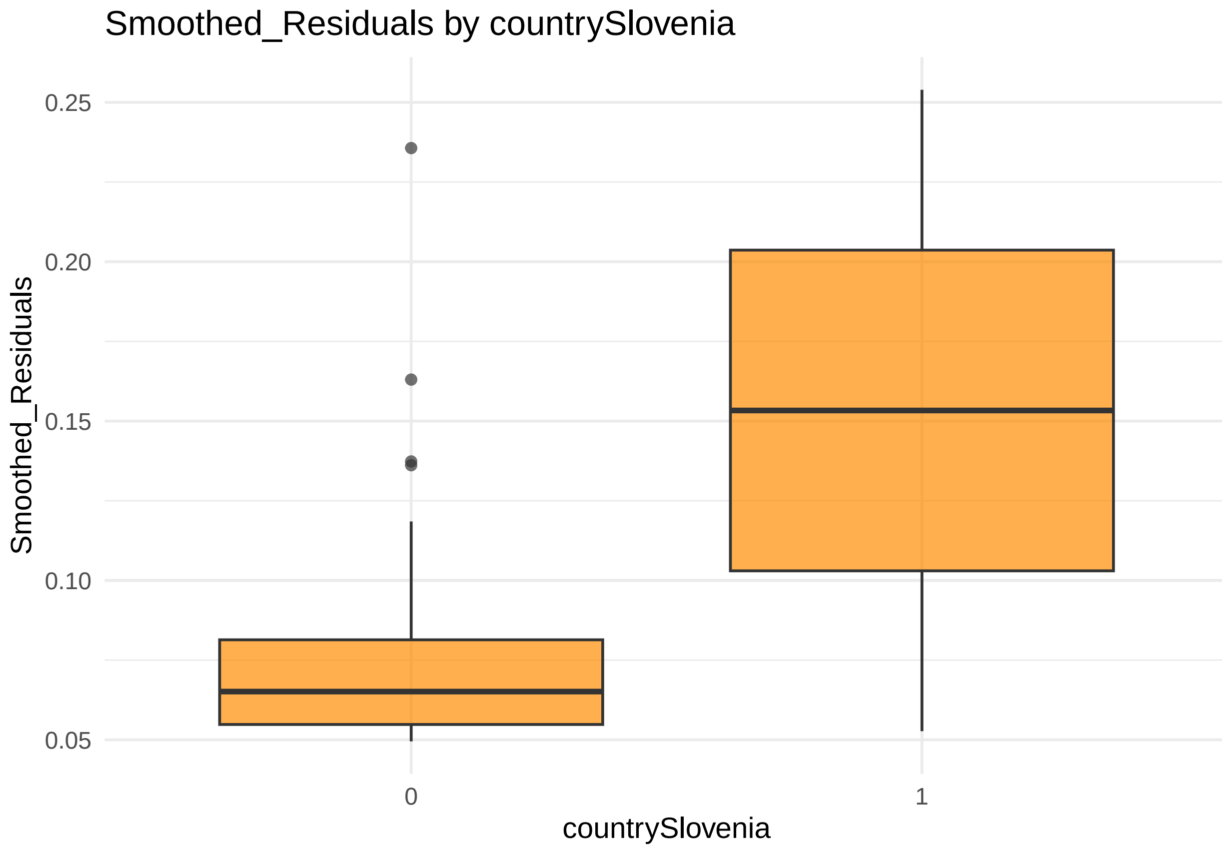

Smoothed_Residuals by countryMalta
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryMalta

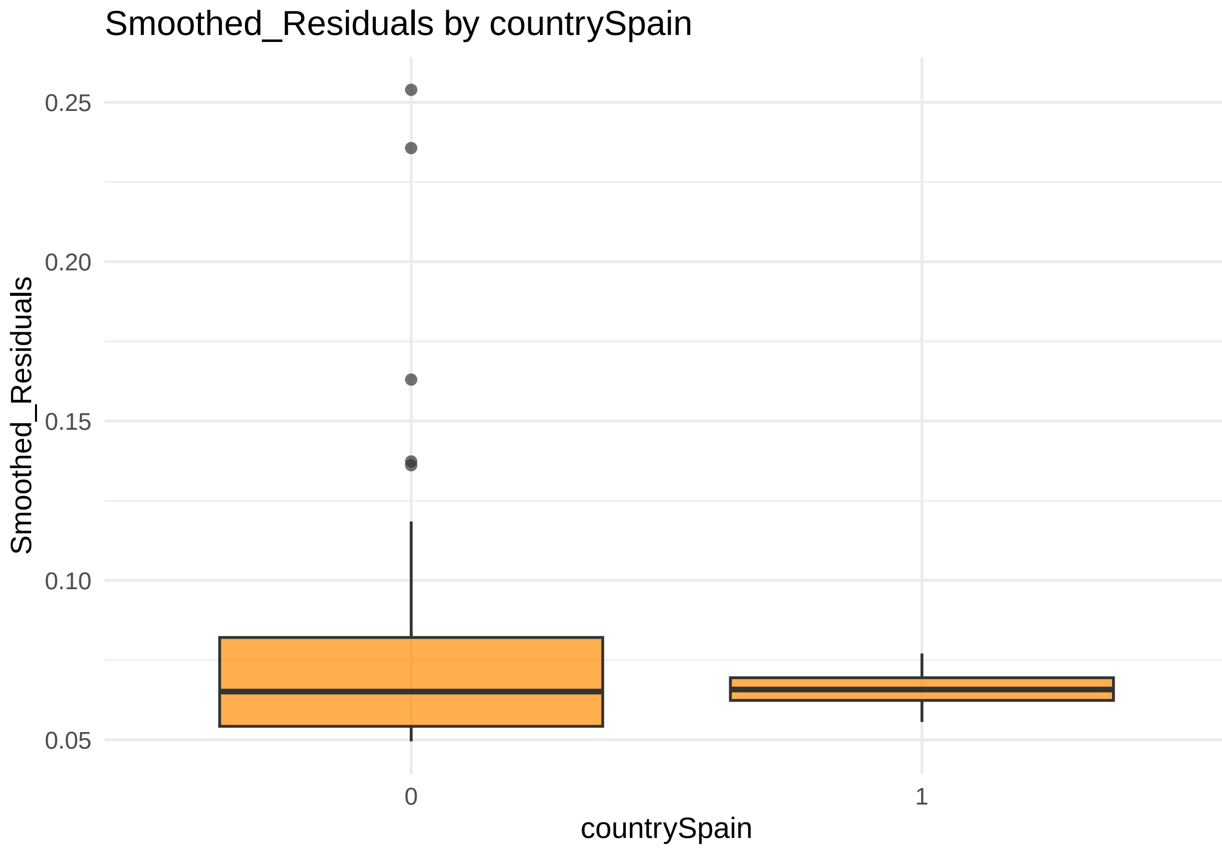

Smoothed_Residuals by countryPoland
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryPoland

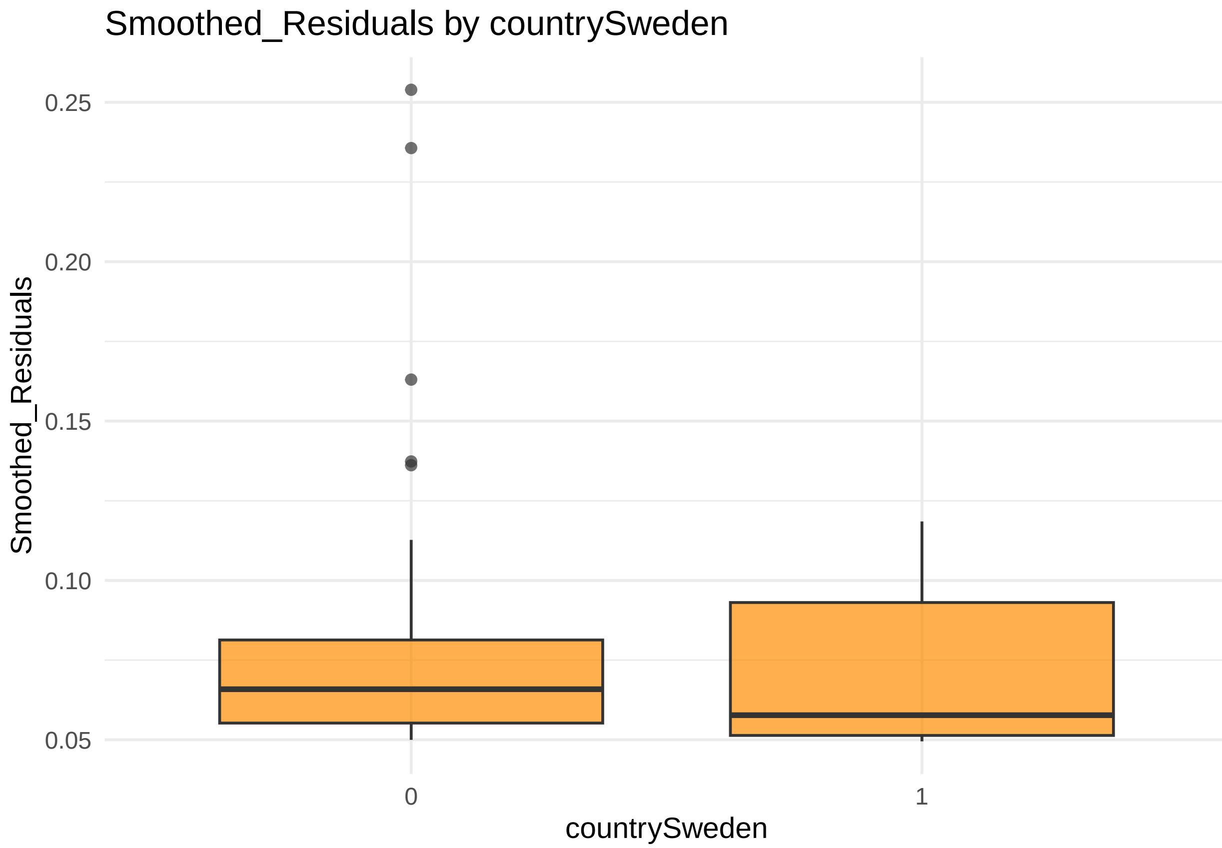

Smoothed_Residuals by countryPortugal
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryPortugal

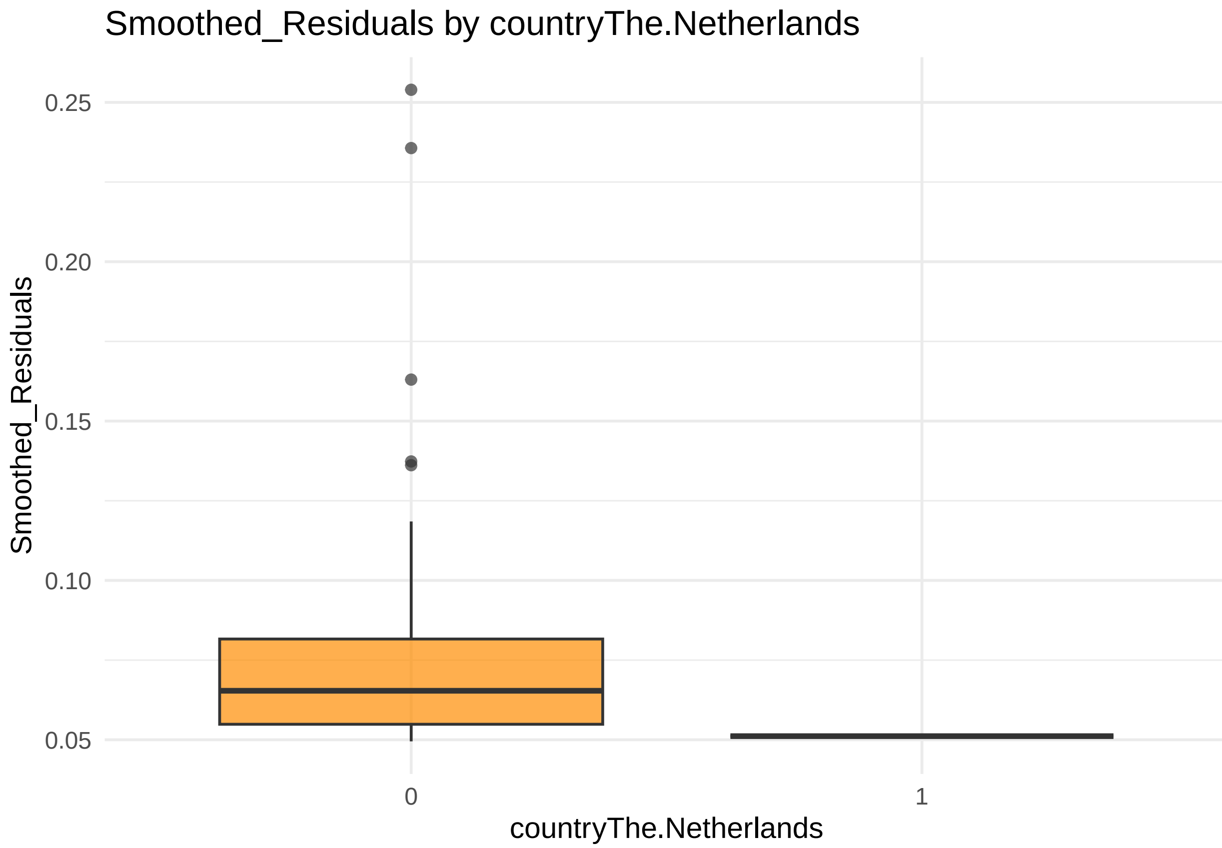

Smoothed_Residuals by countryRomania
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryRomania

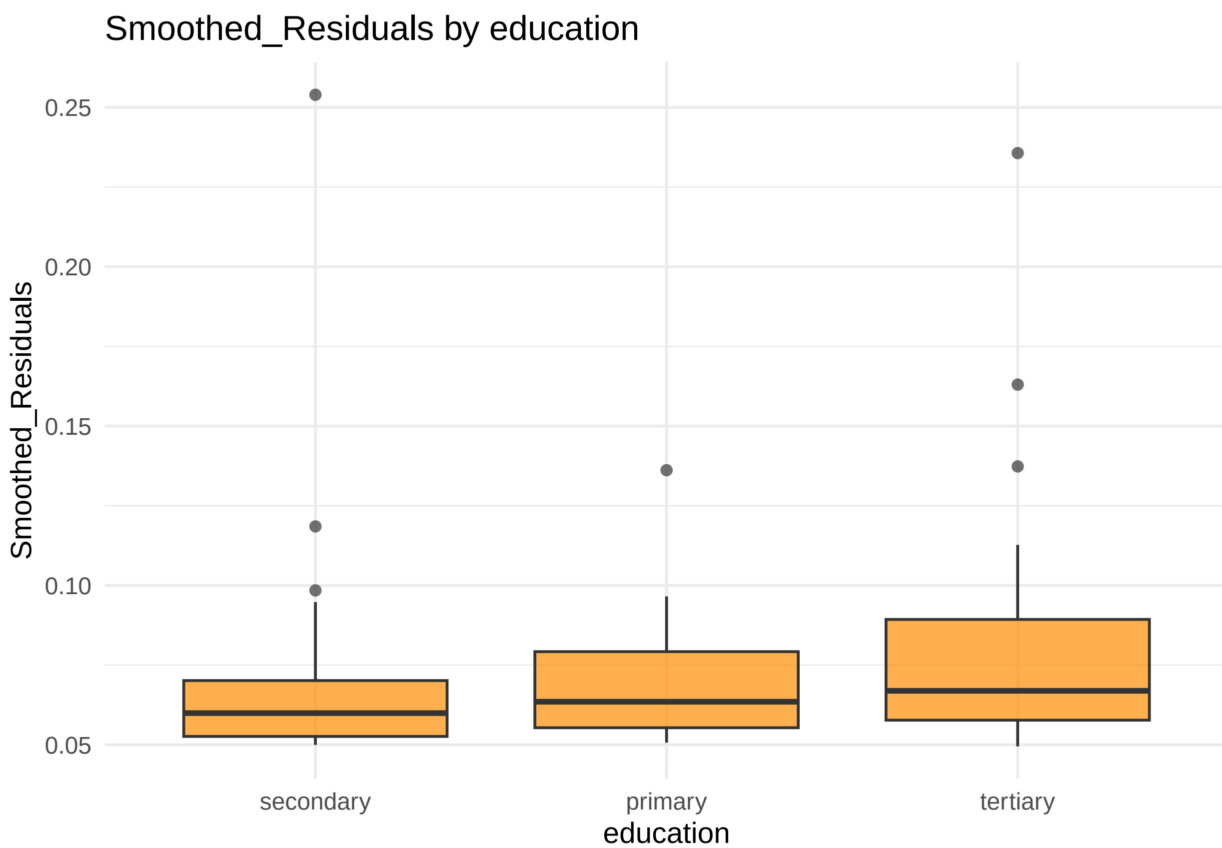

Smoothed_Residuals by countrySlovakia
0.25
0.20
0.15
0.10
0.05
Smoothed_Residuals
0
1
countrySlovakia

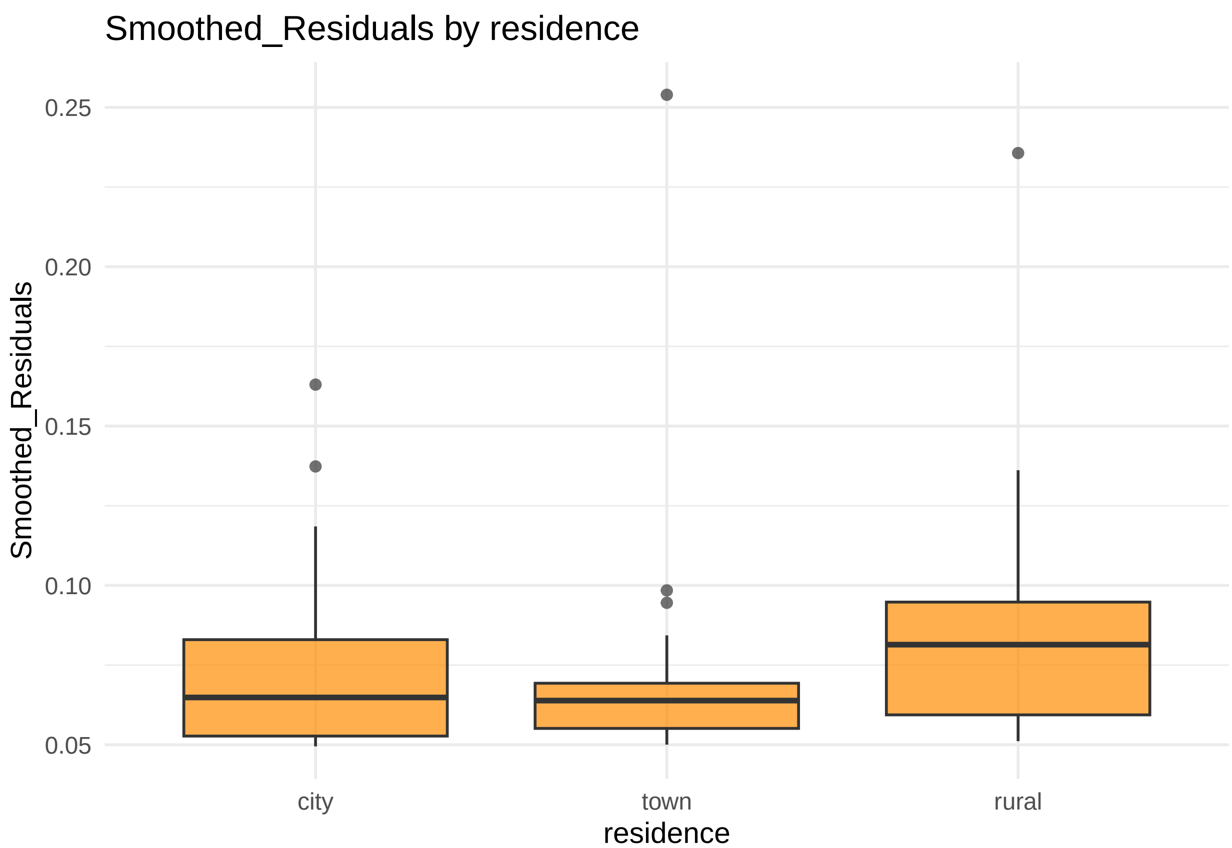

Smoothed_Residuals by countrySlovenia
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countrySlovenia


Smoothed_Residuals by countrySpain
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countrySpain


Smoothed_Residuals by countrySweden
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countrySweden


Smoothed_Residuals by countryThe.Netherlands
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
0
1
countryThe.Netherlands


Smoothed_Residuals by education
Smoothed_Residuals
0.25
0.20
0.15
0.10
0.05
secondary
primary
tertiary
education



Checking the overrepresentation and underrepresentation of any group of people:

```
count_groups <- function(data) {
  for (var in names(data)) {
    x <- data[[var]]

    # Only apply to categorical, character, or binary numeric variables
    is_binary <- is.numeric(x) && length(unique(na.omit(x))) == 2
    is_categorical <- is.factor(x) || is.character(x) || is_binary

    if (is_categorical) {
      cat("----", var, "----\n")
      print(table(x, useNA = "ifany"))
      cat("\n")
    }
  }
}

barplot_srm <- function(data) {
  for (var in names(data)) {
    x <- data[[var]]
```

```r
    # Only process binary, factor, or character variables
    is_binary <- is.numeric(x) && length(unique(na.omit(x))) == 2
    is_categorical <- is.factor(x) || is.character(x) || is_binary

    if (is_categorical) {
      df <- data.frame(group = as.factor(x))

      p <- ggplot(df, aes(x = group)) +
        geom_bar(fill = "steelblue") +
        geom_text(stat = "count", aes(label = ..count..), vjust = -0.3) +
        labs(title = paste("Counts by", var),
             x = var, y = "Count") +
        theme_minimal()

      print(p)
    }
  }
}
```

```r
count_groups(srm_final)
```

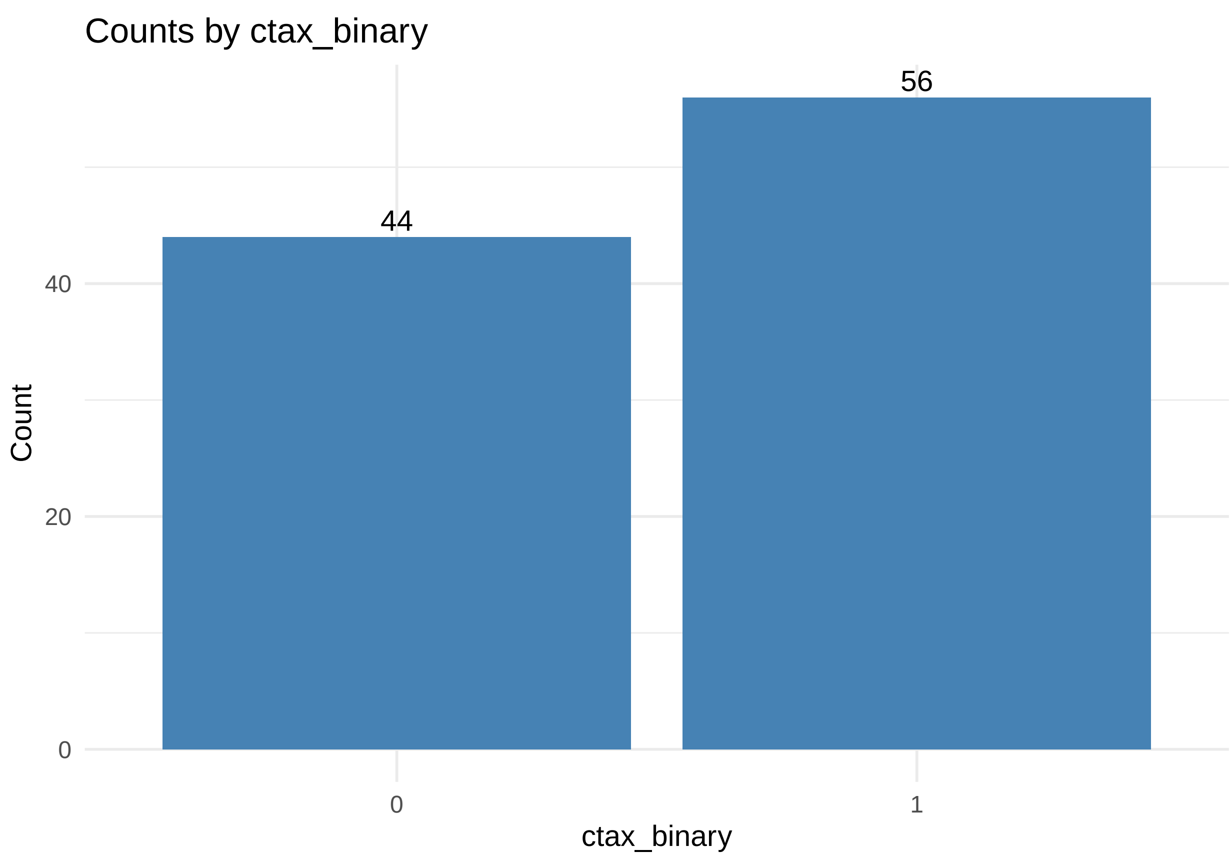
```
## ---- ctax_binary ----
## x
##  0  1
## 44 56
##
## ---- trust ----
## x
## No confident at all No really confident    Rather confident      Very confident
##                  34                  18                  32                  16
##
## ---- any_cc_last2year_factor ----
## x
##  0  1
## 47 53
##
## ---- regional_heterogeneity ----
## x
##  0  1
## 19 81
##
## ---- gender ----
## x
## female   male
##     39     61
##
## ---- has_childrenyes ----
## x
##  0  1
## 58 42
##
## ---- countryBelgium ----
## x
```

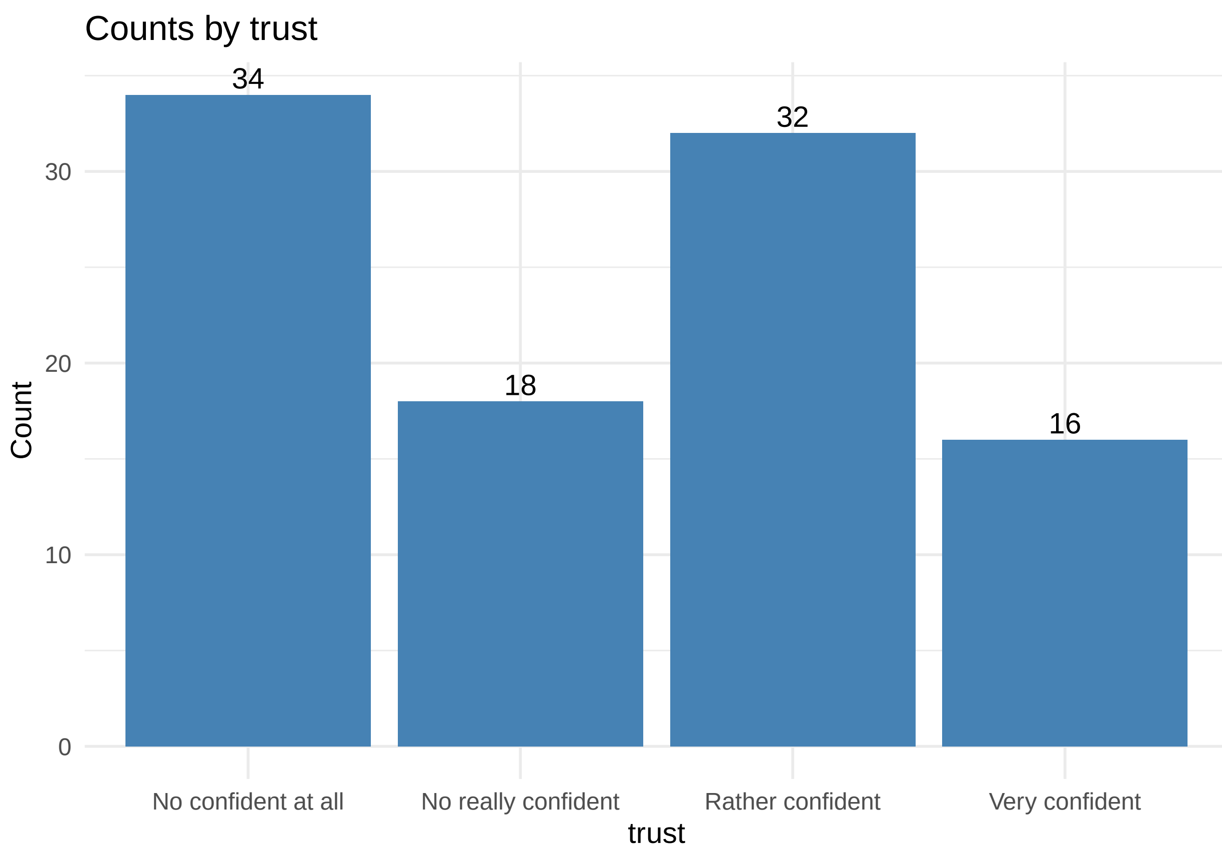
```
##  0  1
## 97  3
##
## ---- countryBulgaria ----
## x
##  0  1
## 99  1
##
## ---- countryCroatia ----
## x
##  0  1
## 93  7
##
## ---- countryCyprus ----
## x
##  0  1
## 99  1
##
## ---- countryCzech.Republic ----
## x
##  0  1
## 99  1
##
## ---- countryDenmark ----
## x
##  0  1
## 91  9
##
## ---- countryFinland ----
## x
##  0  1
## 94  6
##
## ---- countryFrance ----
## x
##  0  1
## 98  2
##
## ---- countryGermany ----
## x
##  0  1
## 88 12
##
## ---- countryGreece ----
## x
##  0  1
## 99  1
##
## ---- countryHungary ----
## x
##  0  1
## 97  3
##
## ---- countryIreland ----
```

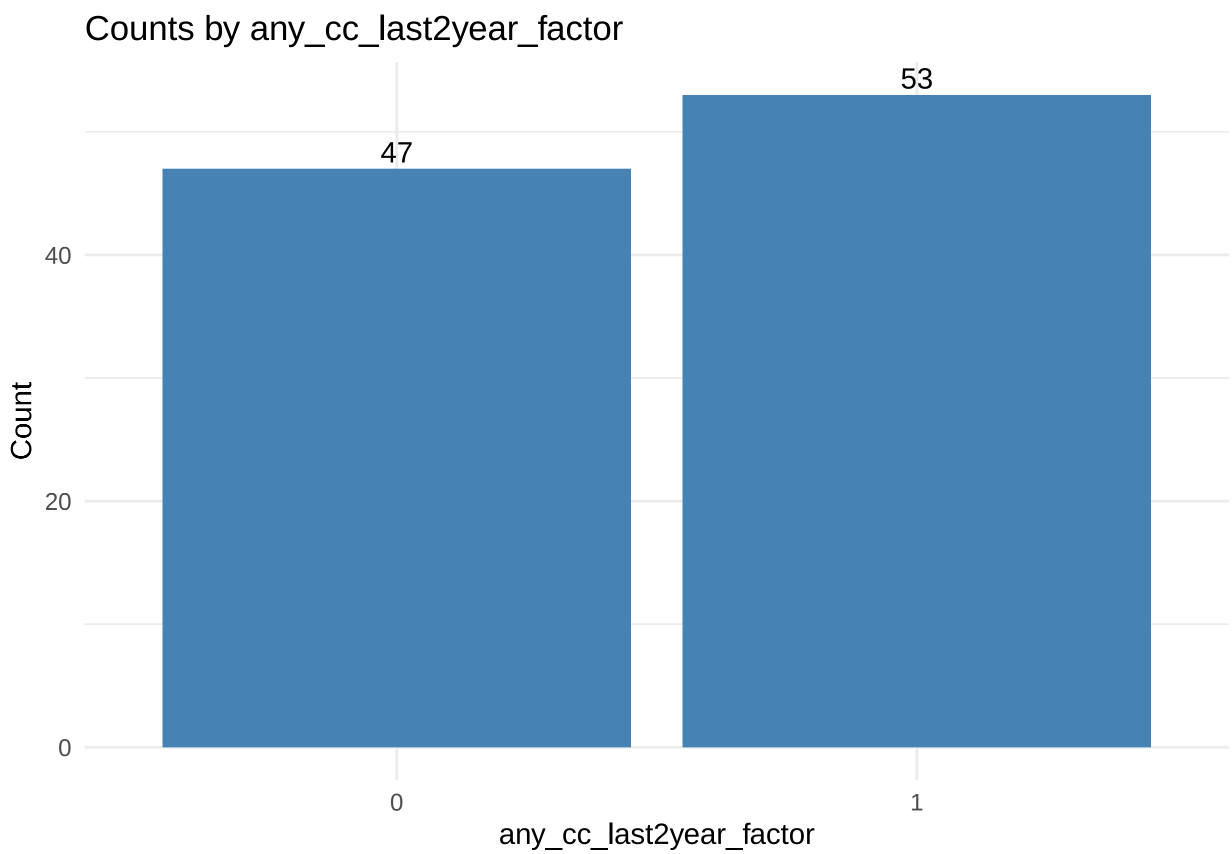
```
## x
##  0  1
## 98  2
##
## ---- countryLatvia ----
## x
##  0  1
## 99  1
##
## ---- countryLuxembourg ----
## x
##  0  1
## 97  3
##
## ---- countryMalta ----
## x
##  0  1
## 99  1
##
## ---- countryPoland ----
## x
##  0  1
## 98  2
##
## ---- countryPortugal ----
## x
##  0  1
## 99  1
##
## ---- countryRomania ----
## x
##  0  1
## 96  4
##
## ---- countrySlovakia ----
## x
##  0  1
## 95  5
##
## ---- countrySlovenia ----
## x
##  0  1
## 98  2
##
## ---- countrySpain ----
## x
##  0  1
## 96  4
##
## ---- countrySweden ----
## x
##  0  1
## 87 13
##
```

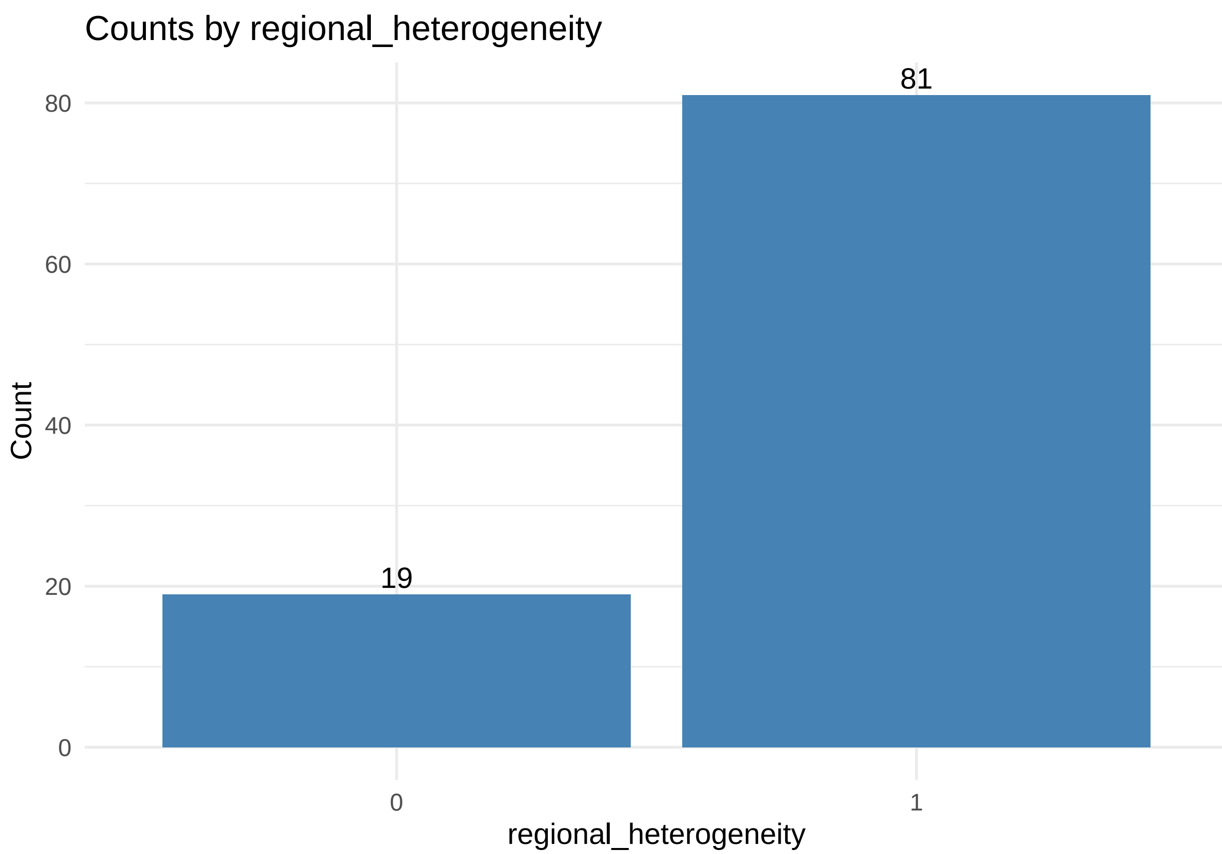
```
## ---- countryThe.Netherlands ----
## x
##  0  1
## 99  1
##
## ---- education ----
## x
## secondary   primary  tertiary
##        41        12        47
##
## ---- residence ----
## x
##  city  town rural
##    48    35    17
```

```
barplot_srm(srm_final)
```

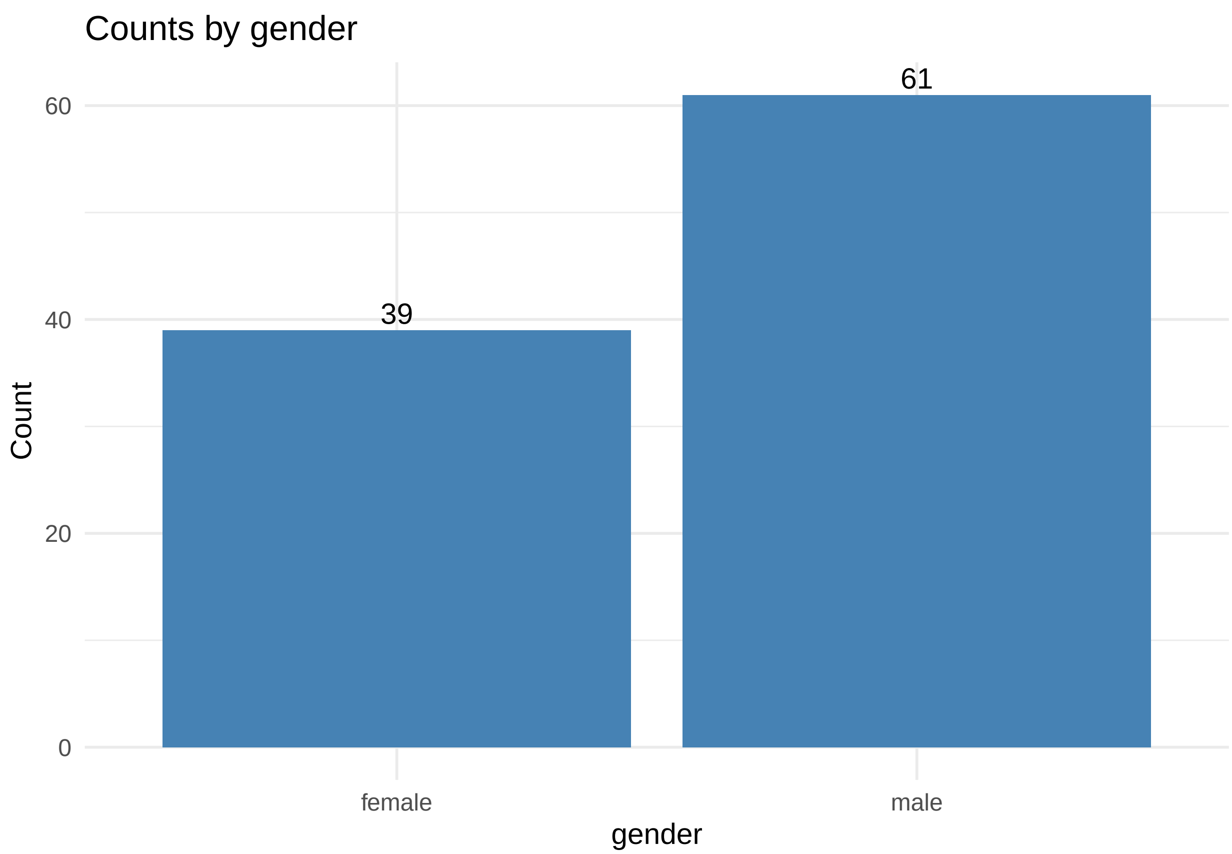


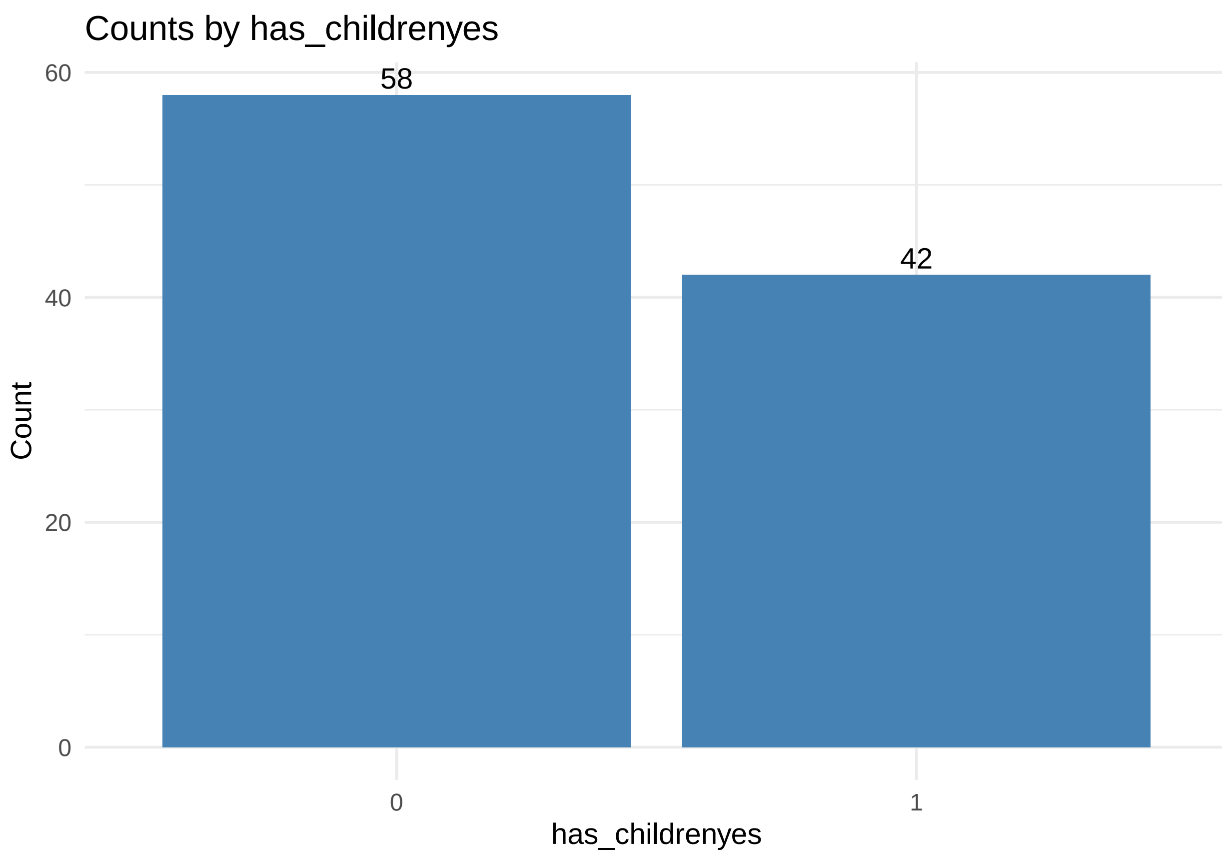

Counts by trust
34
32
18
16
30
20
10
0
Count
No confident at all
No really confident
Rather confident
Very confident
trust

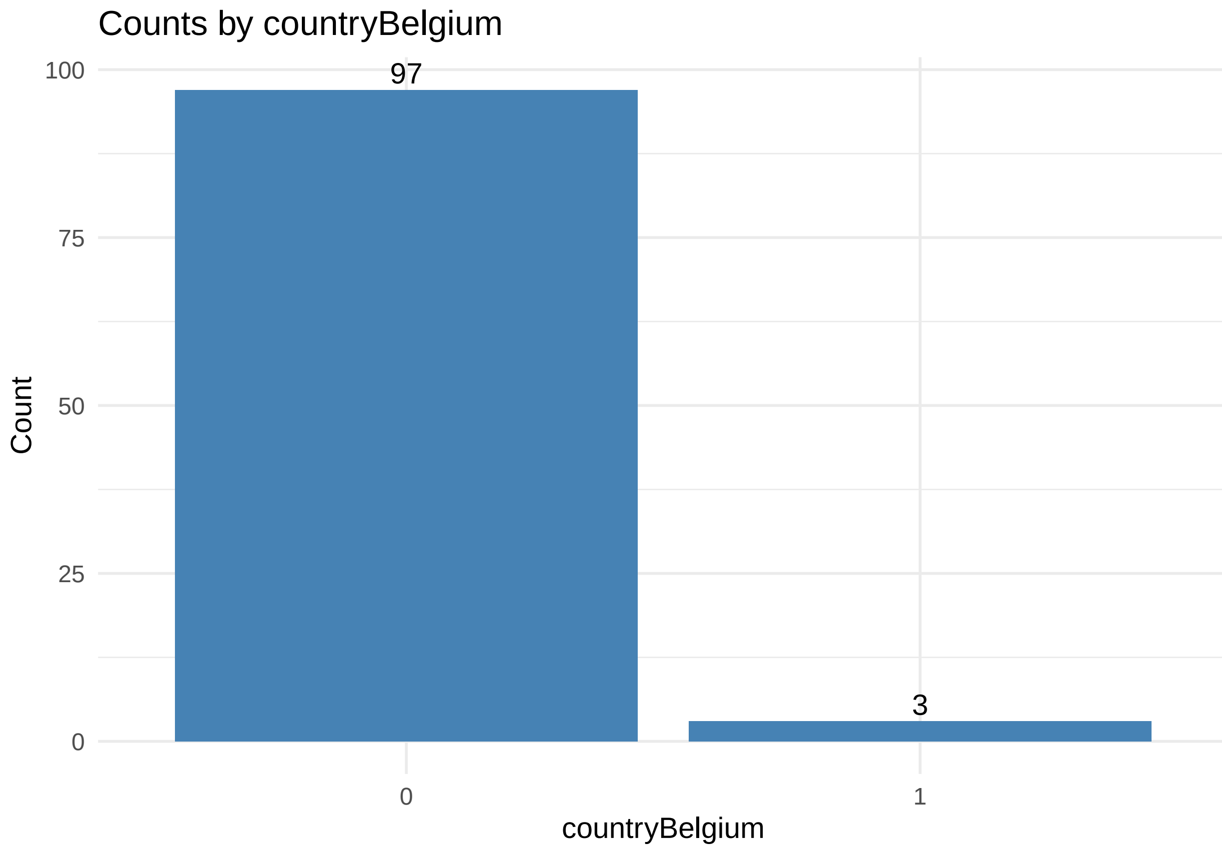

Counts by any_cc_last2year_factor
53
47
40
20
0
Count
0
1
any_cc_last2year_factor



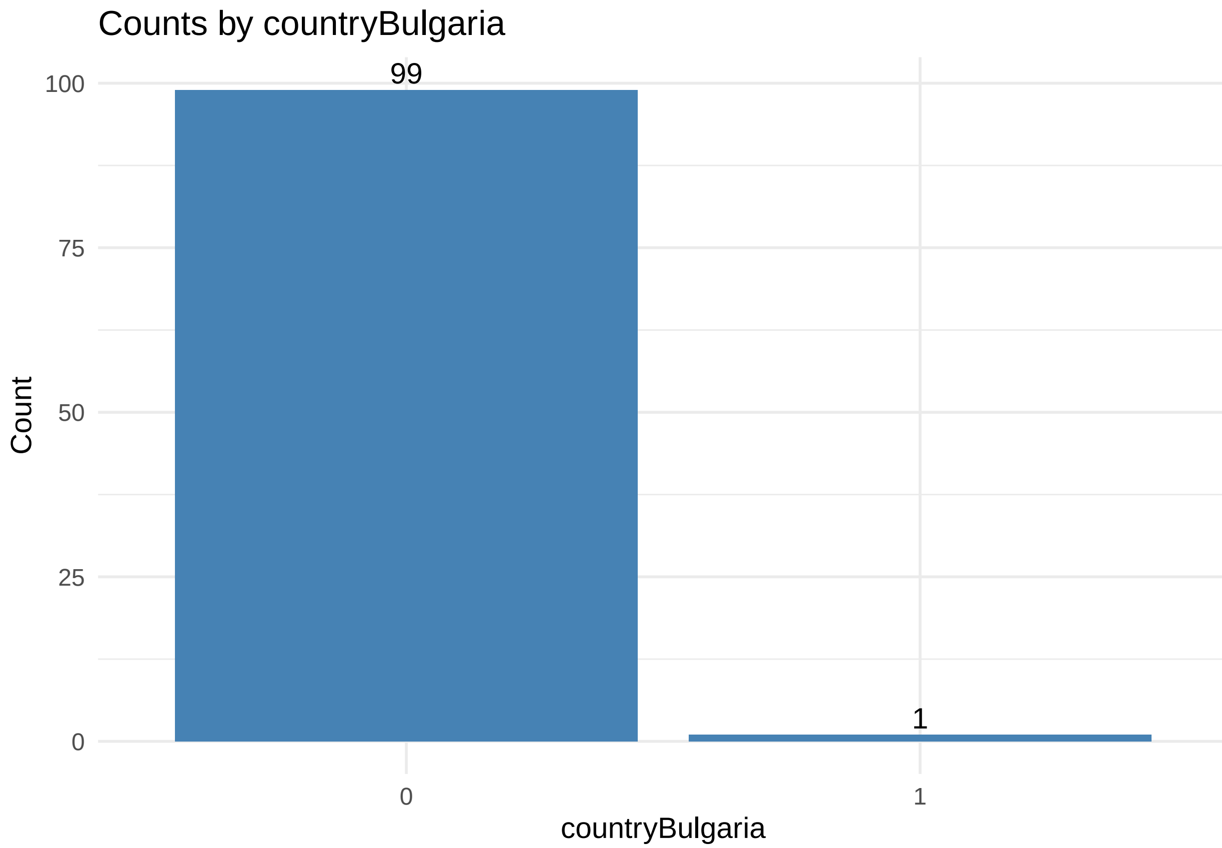
Counts by regional_heterogeneity
81
19
80
60
40
20
0
Count
0
1
regional_heterogeneity

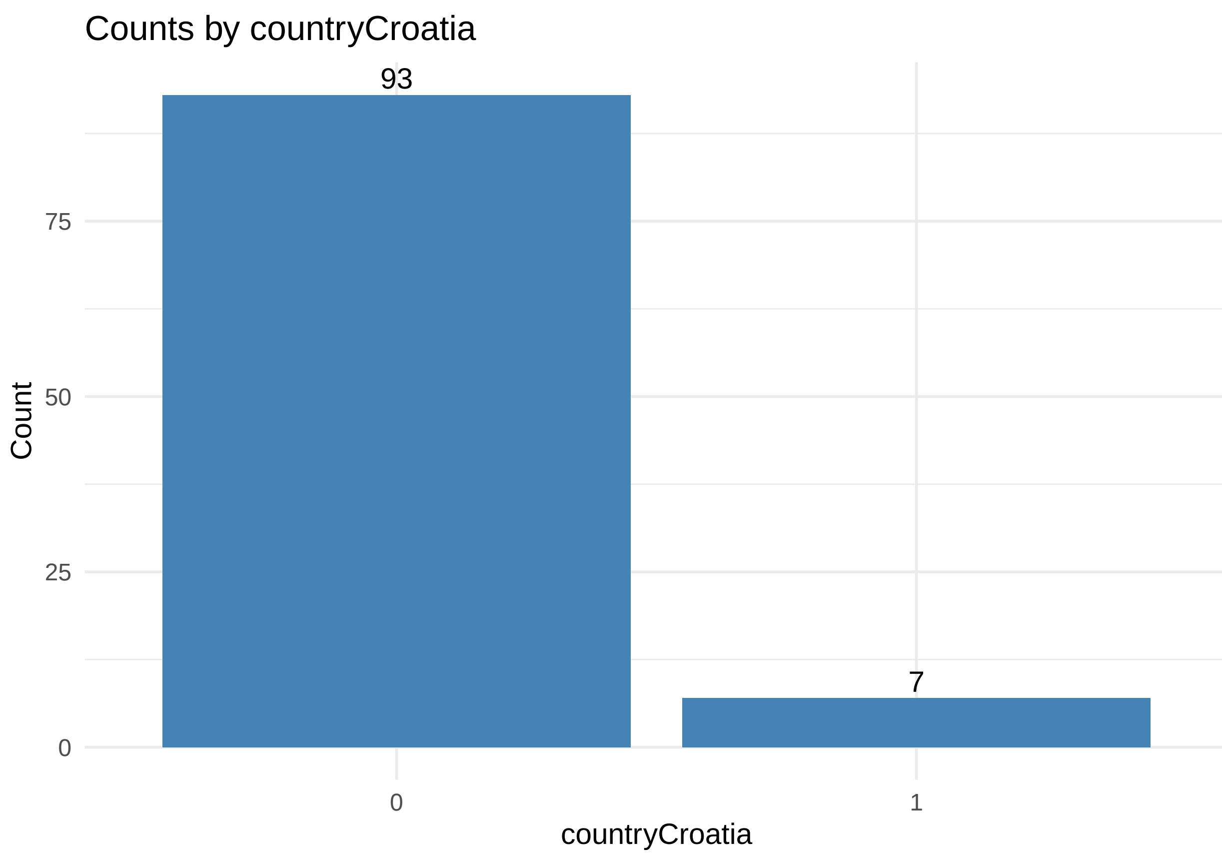

Counts by gender
61
39
60
40
20
0
Count
female
male
gender

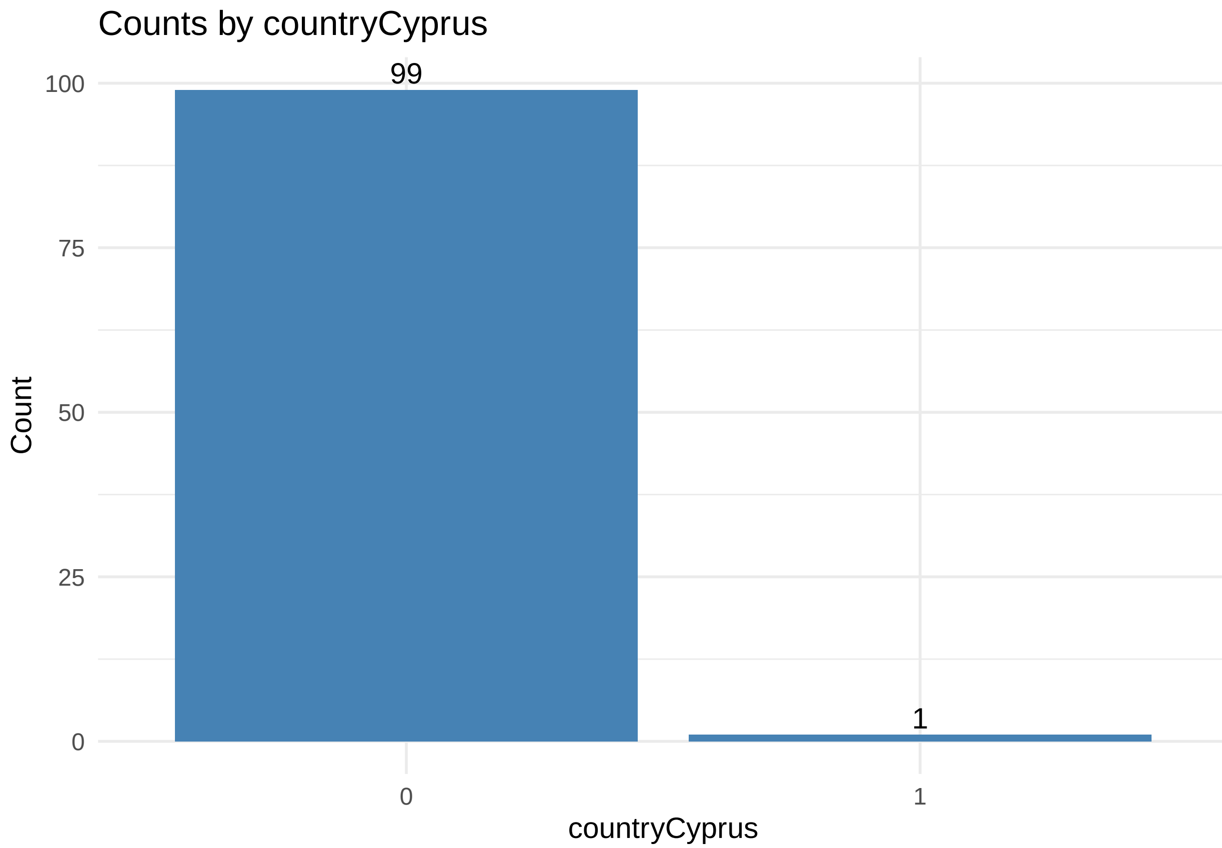

Counts by has_childrenyes
60
40
20
0
58
42
Count
0
1
has_childrenyes



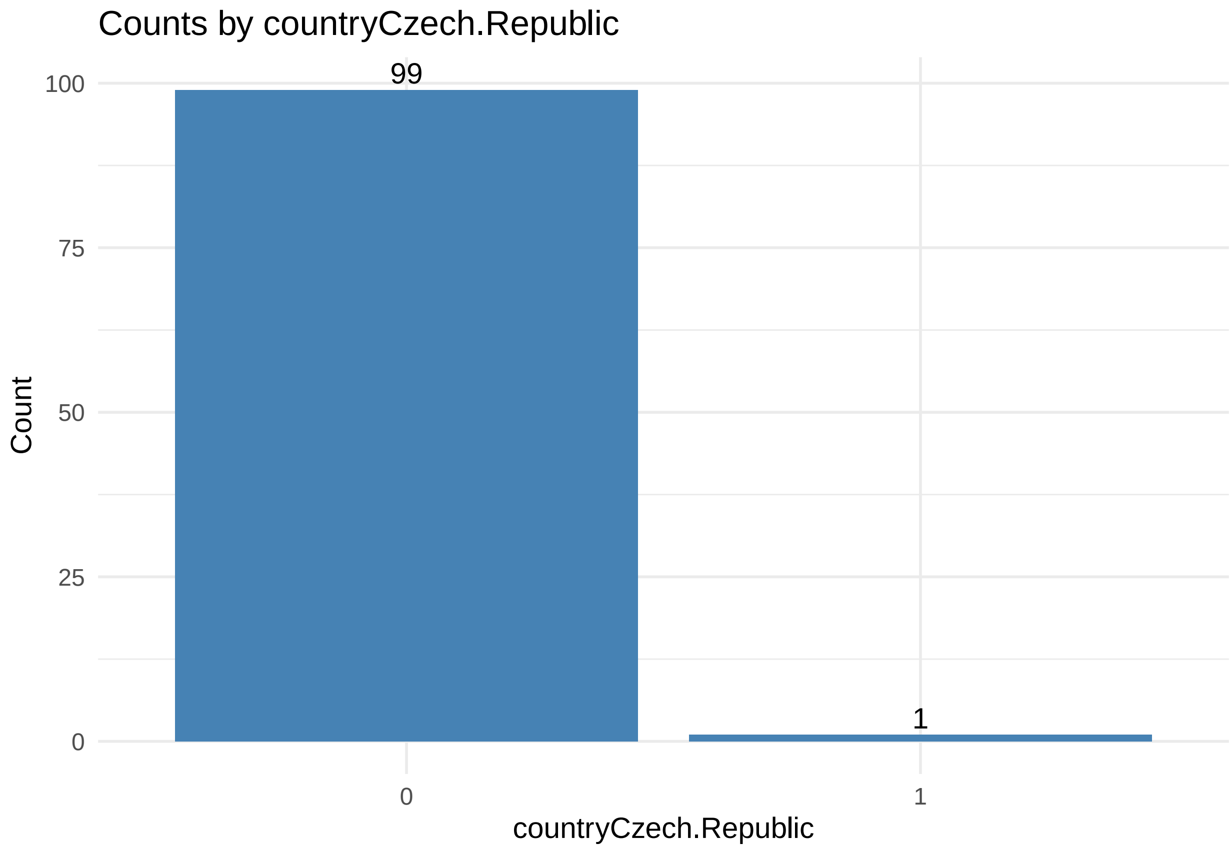
Counts by countryBelgium
Count
countryBelgium
100
75
50
25
0
97
3
0
1

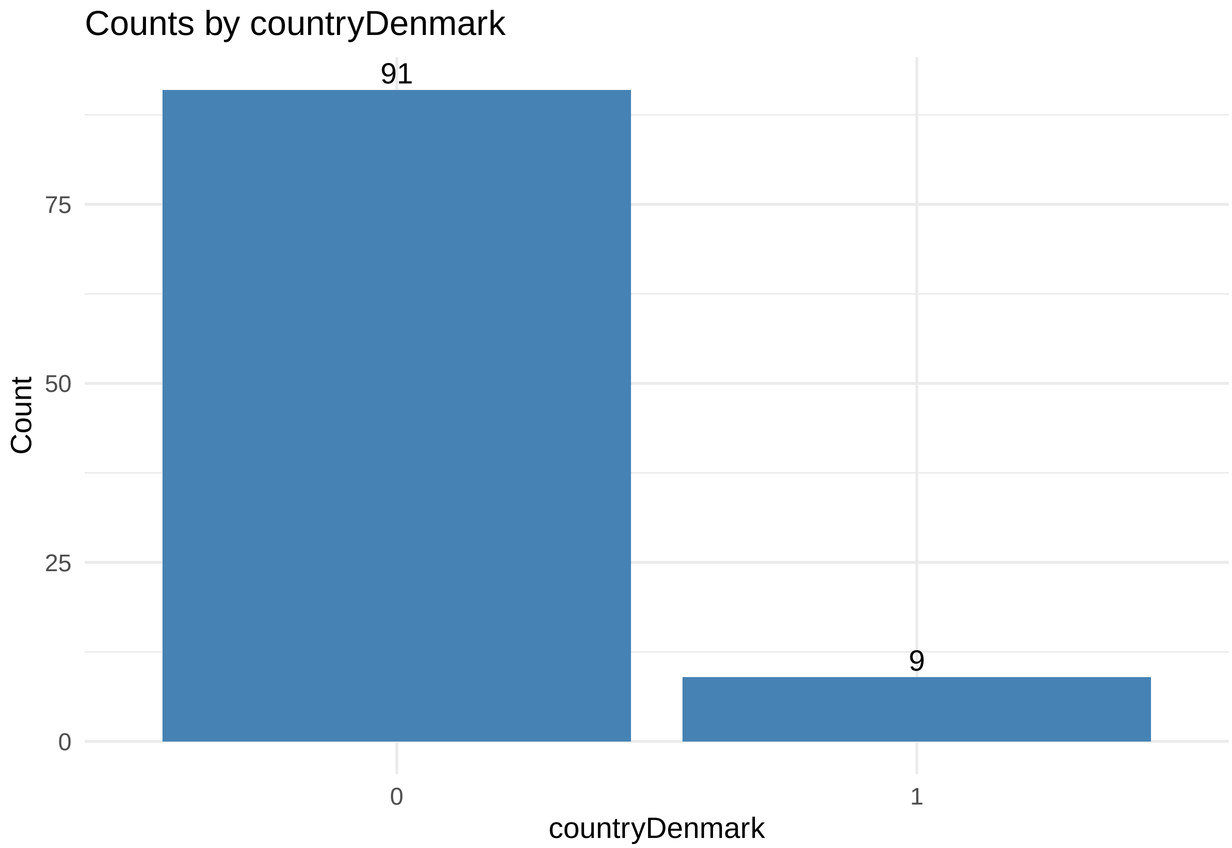

Counts by countryBulgaria
99
1
100
75
50
25
0
Count
0
1
countryBulgaria

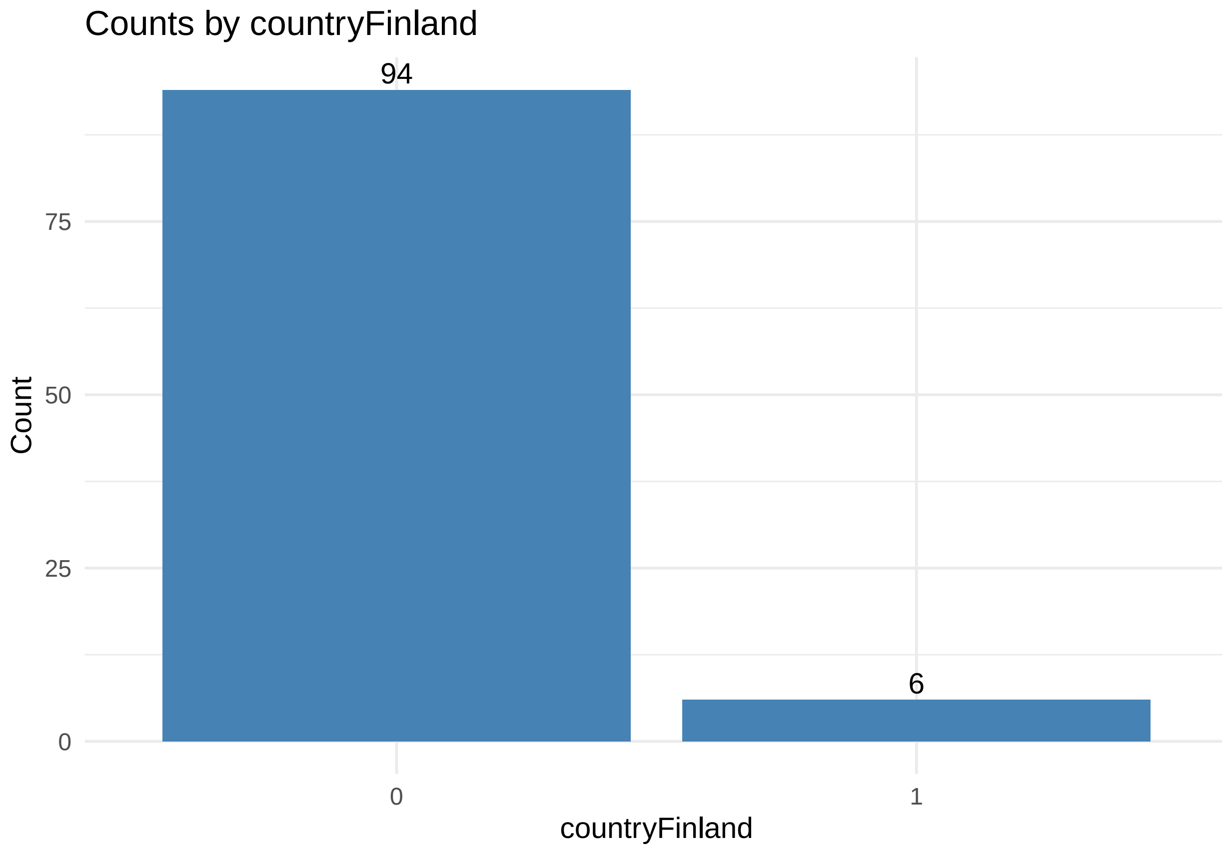

Counts by countryCroatia
93
7
75
50
25
0
Count
0
1
countryCroatia

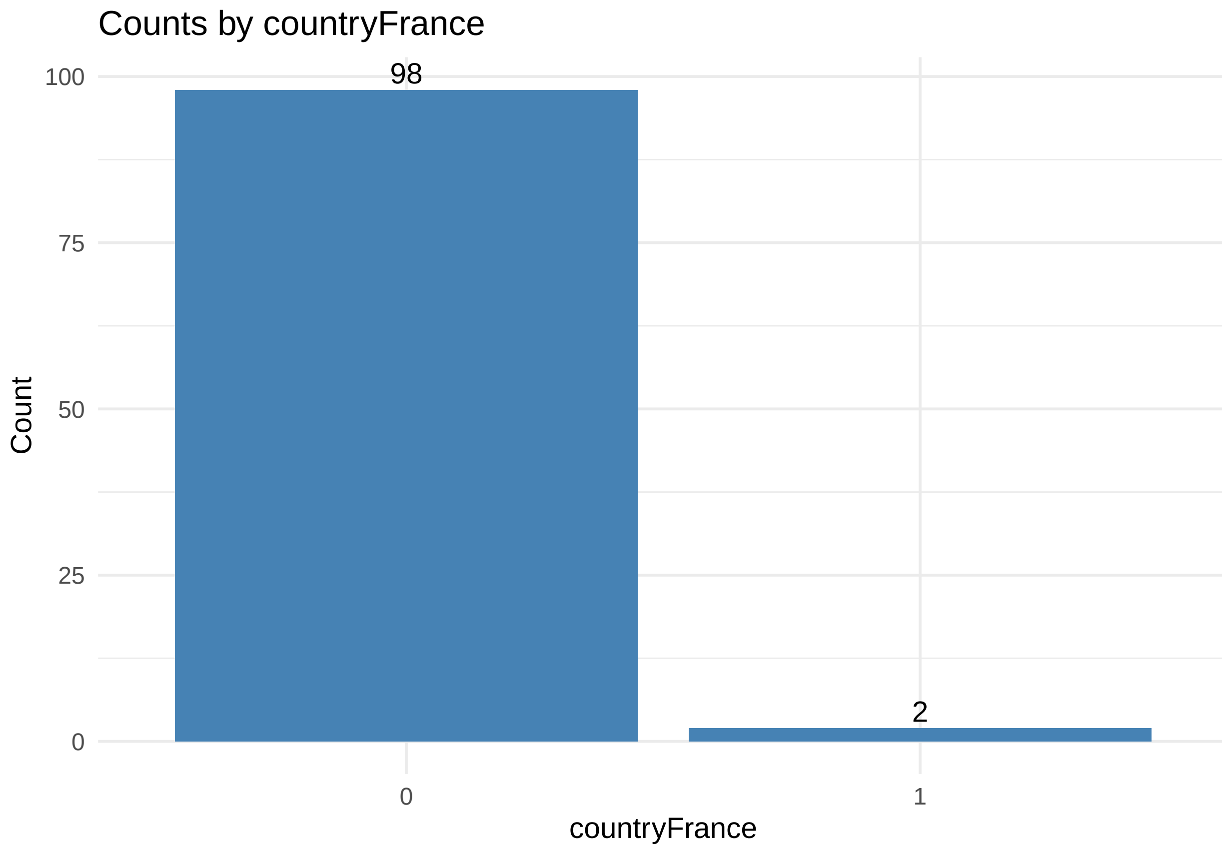


Counts by countryCyprus
99
1
100
75
50
25
0
Count
0
1
countryCyprus

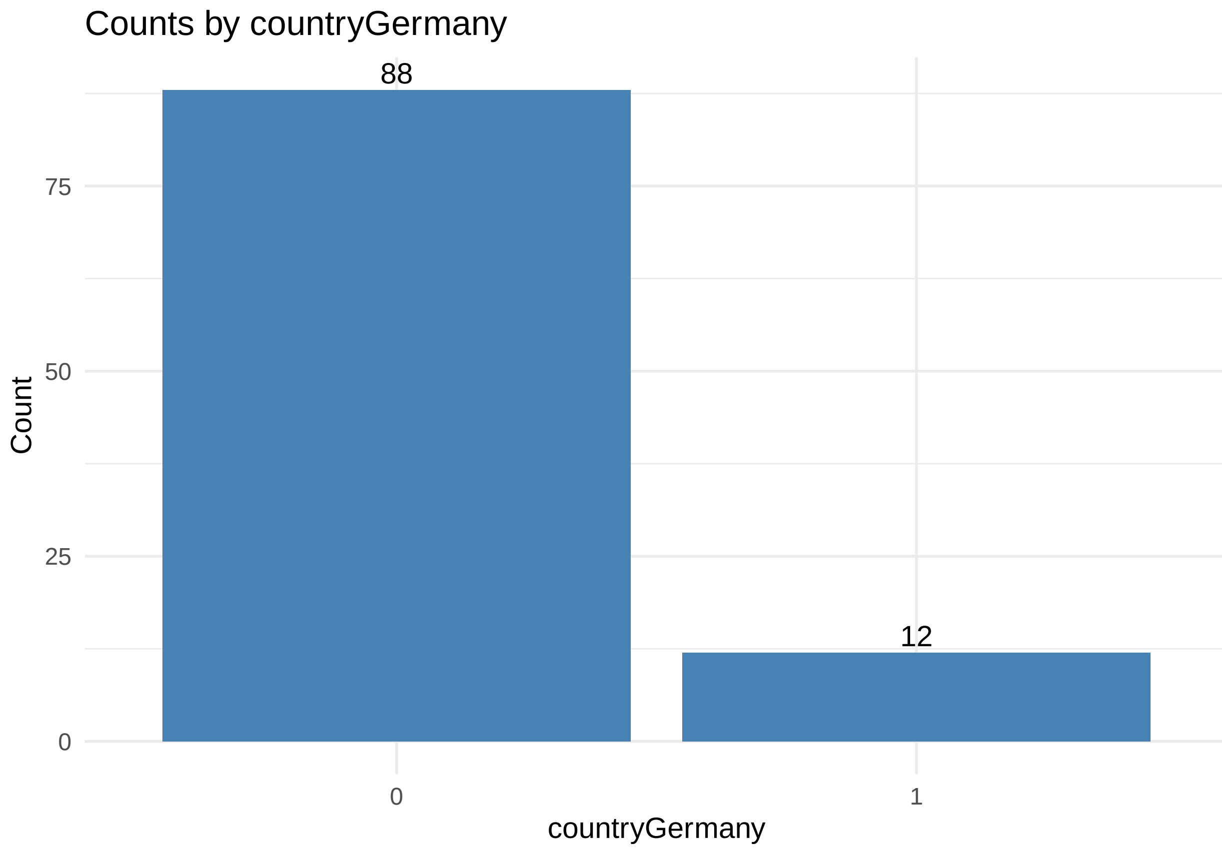

Counts by countryCzech.Republic
99
1
100
75
50
25
0
Count
0
1
countryCzech.Republic

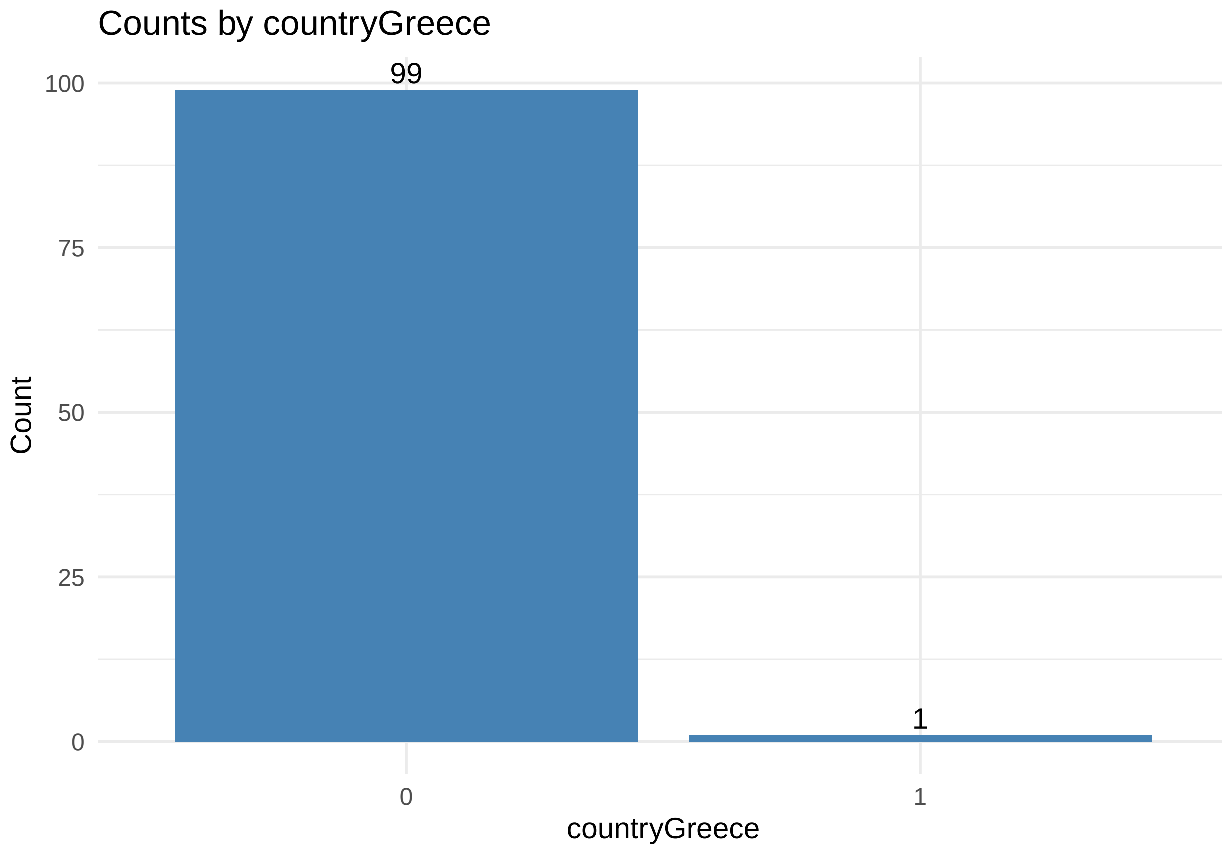

Counts by countryDenmark
91
9
75
50
25
0
Count
0
1
countryDenmark

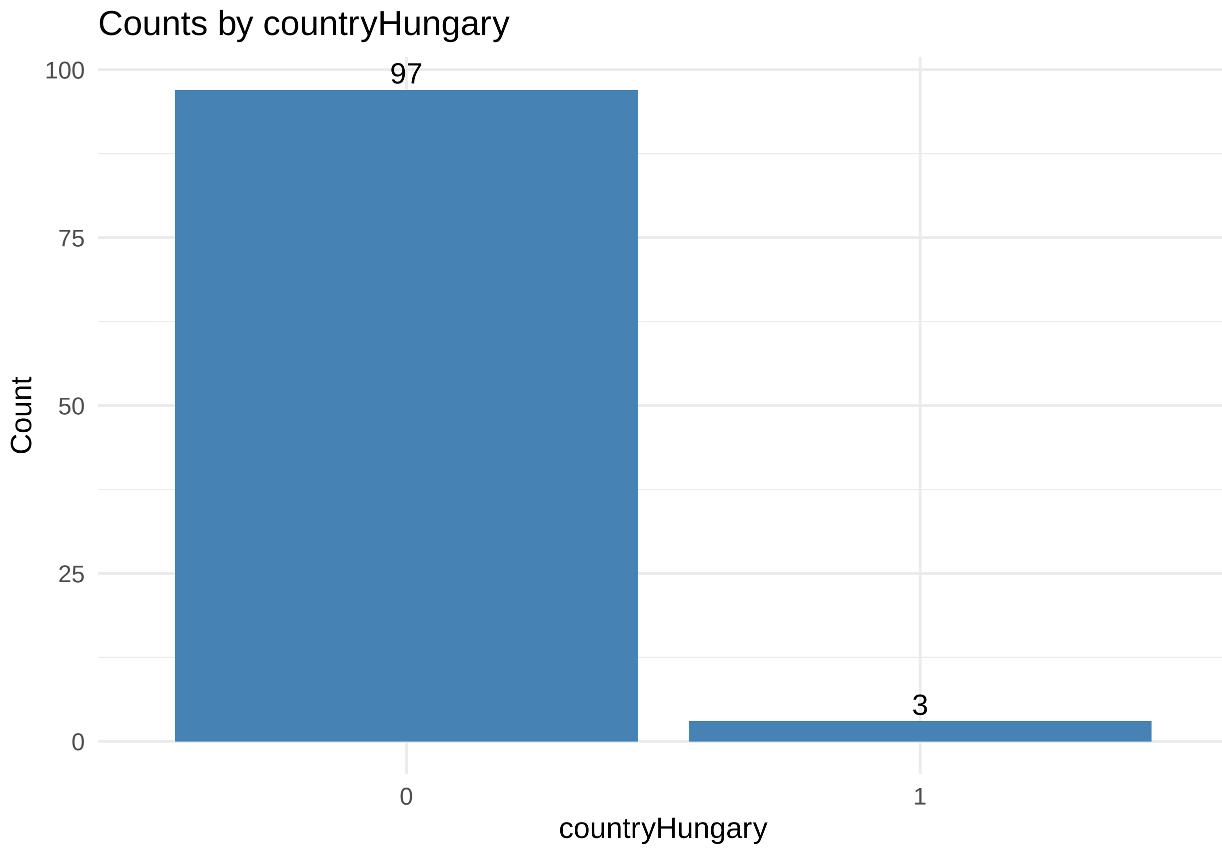

Counts by countryFinland
94
6
75
50
25
0
Count
0
1
countryFinland

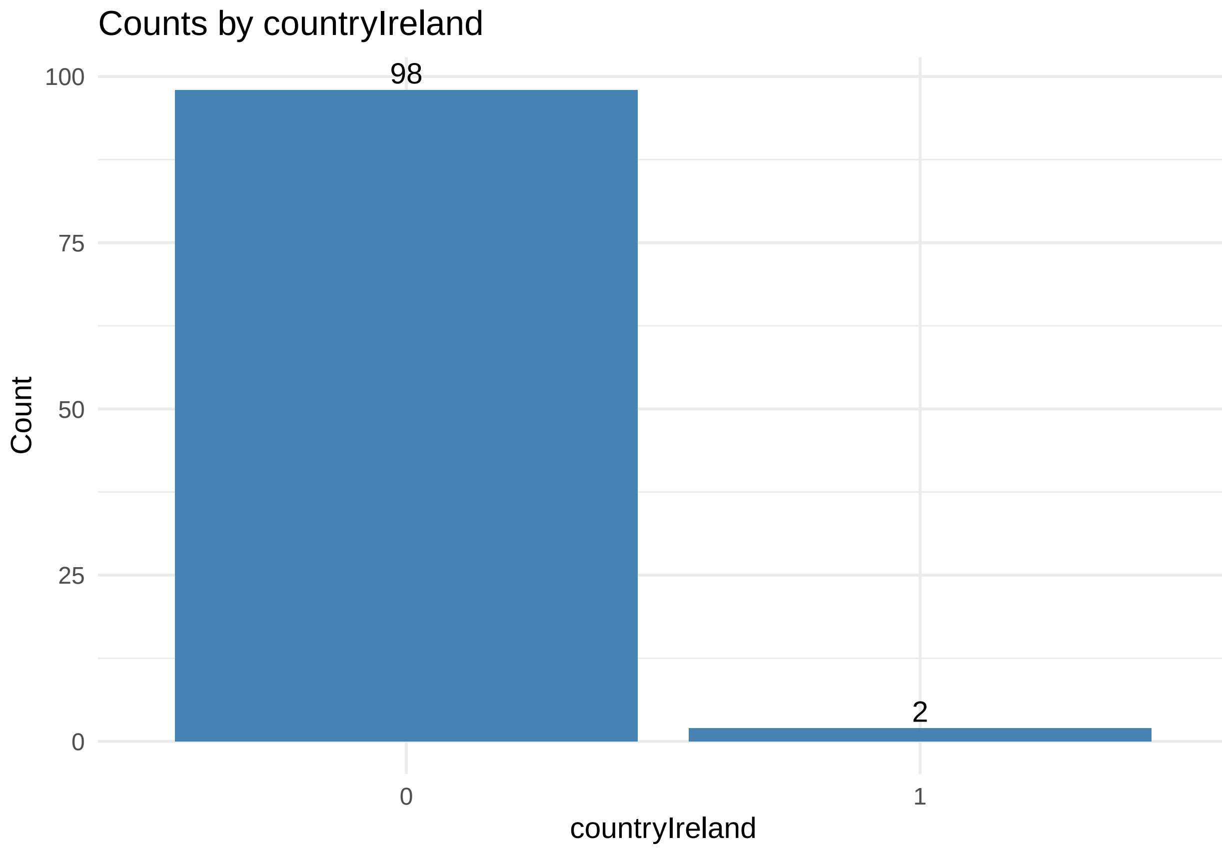


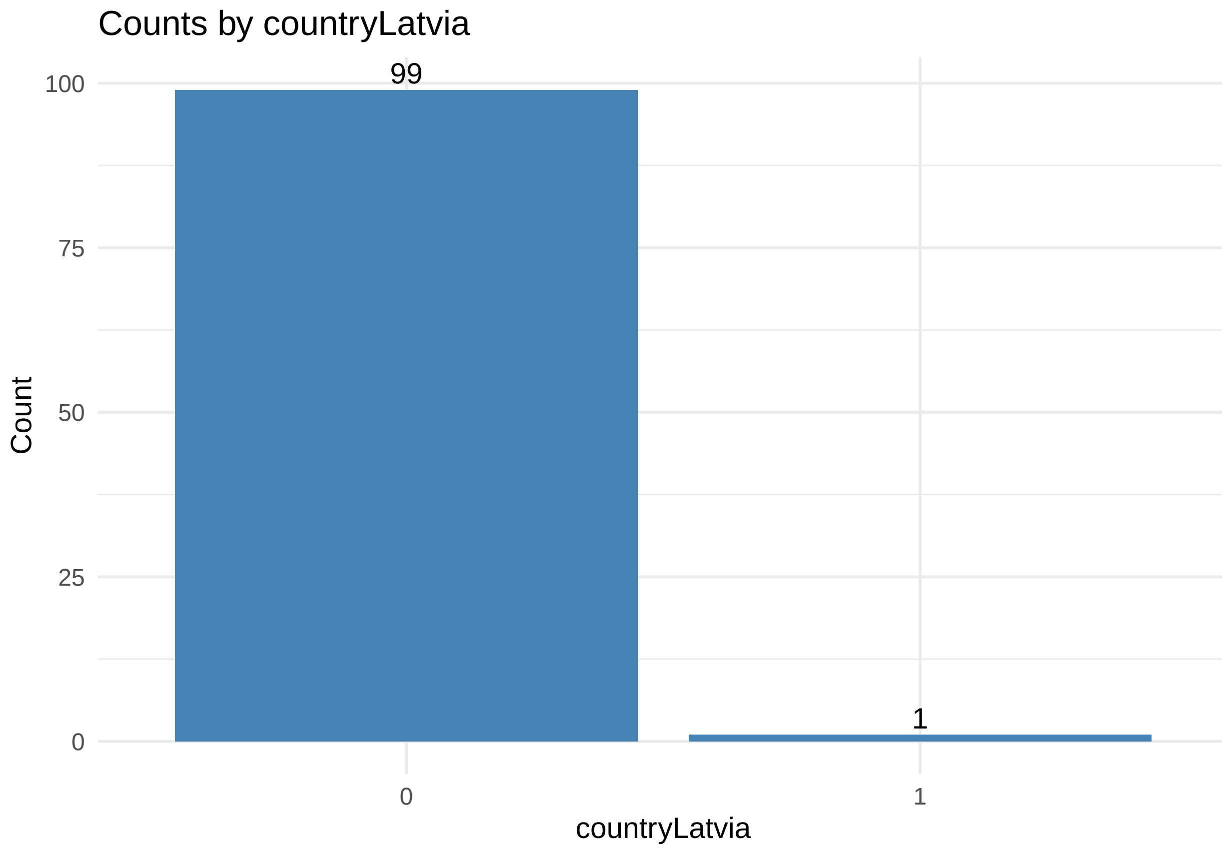
Counts by countryFrance
98
2
100
75
50
25
0
Count
0
1
countryFrance

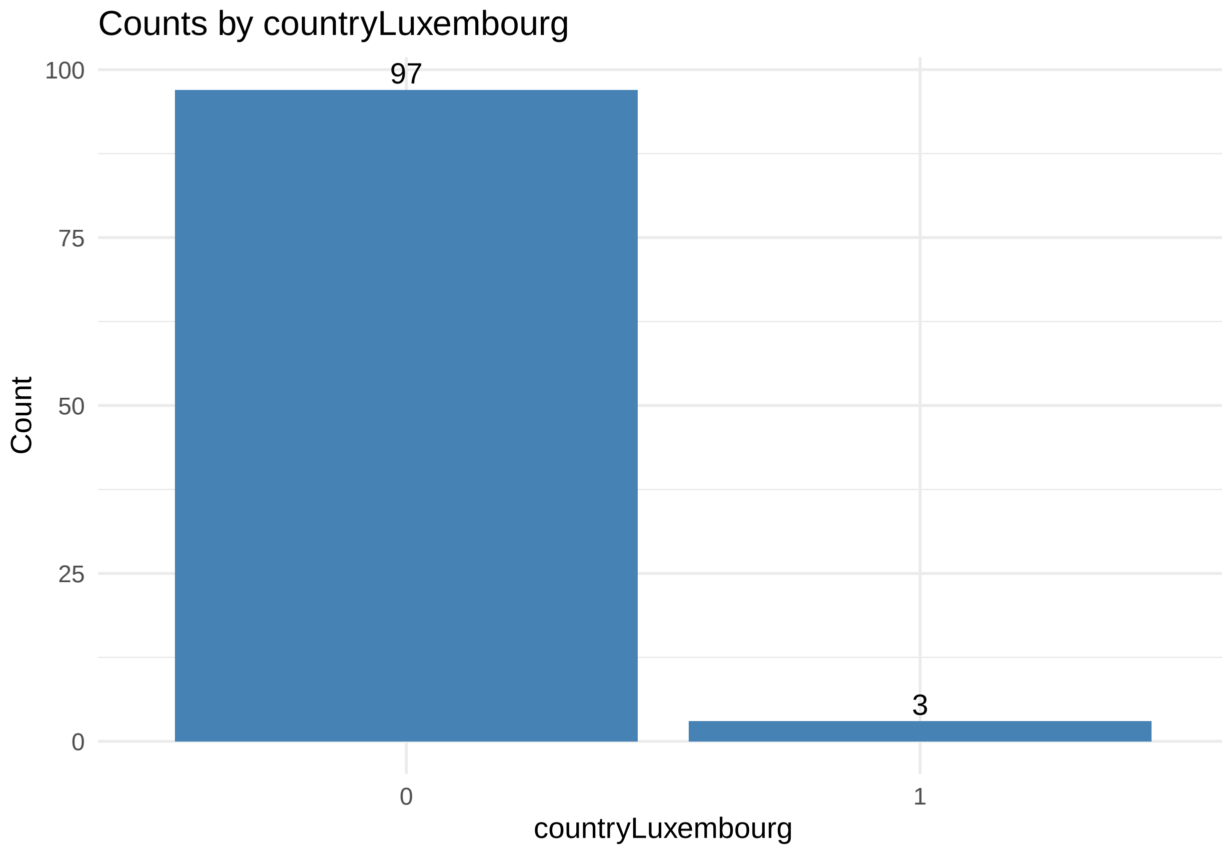

Counts by countryGermany
88
12
75
50
25
0
Count
0
1
countryGermany

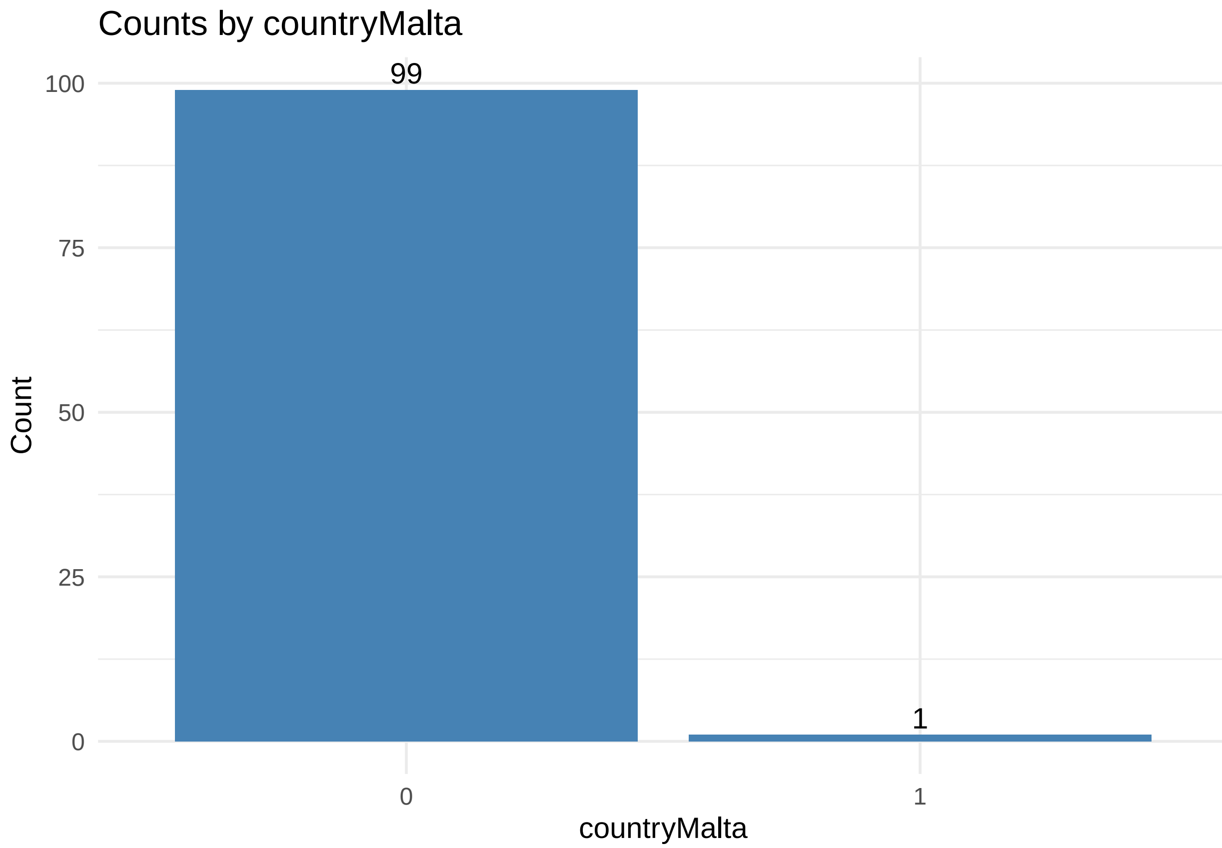

Counts by countryGreece
99
1
Count
100
75
50
25
0
0
1
countryGreece

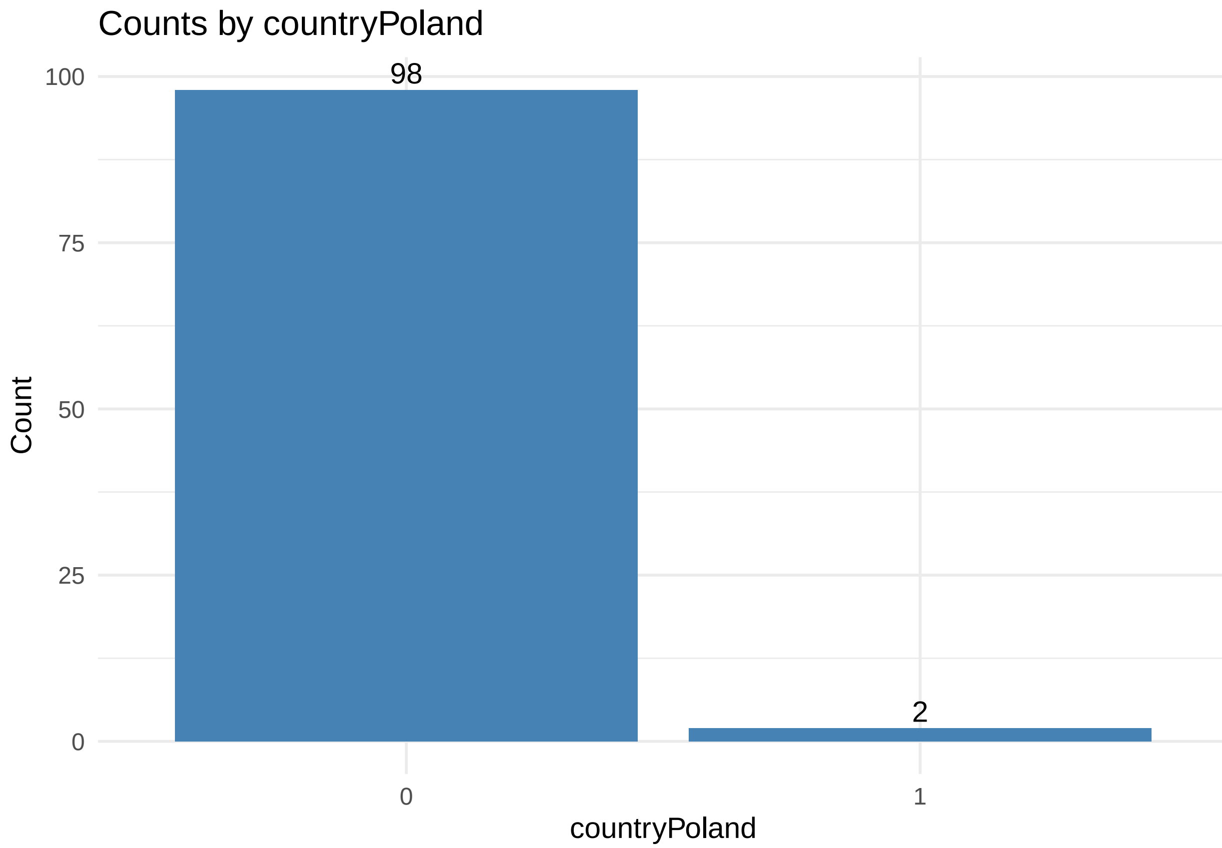

Counts by countryHungary
Count
100
75
50
25
0
97
3
0
1
countryHungary

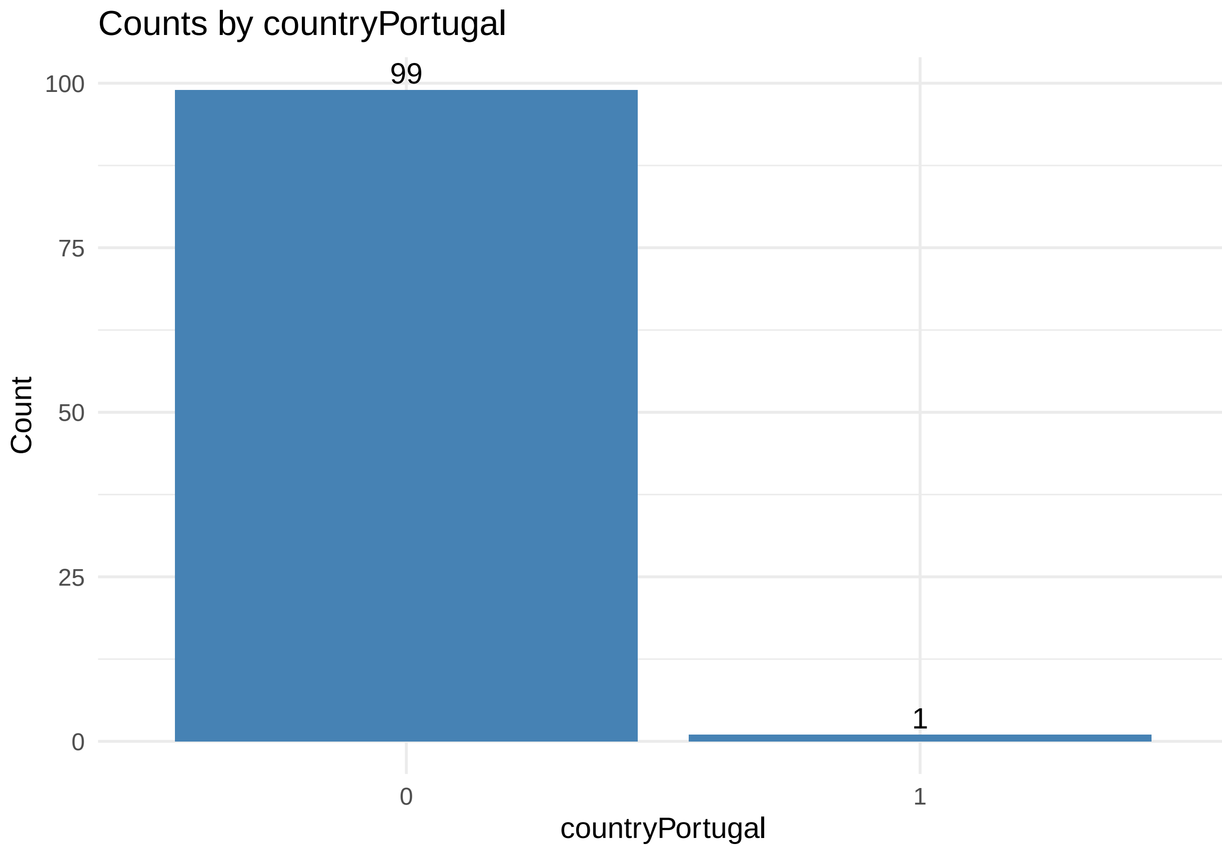

Counts by countryIreland
98
2
100
75
50
25
0
Count
0
1
countryIreland

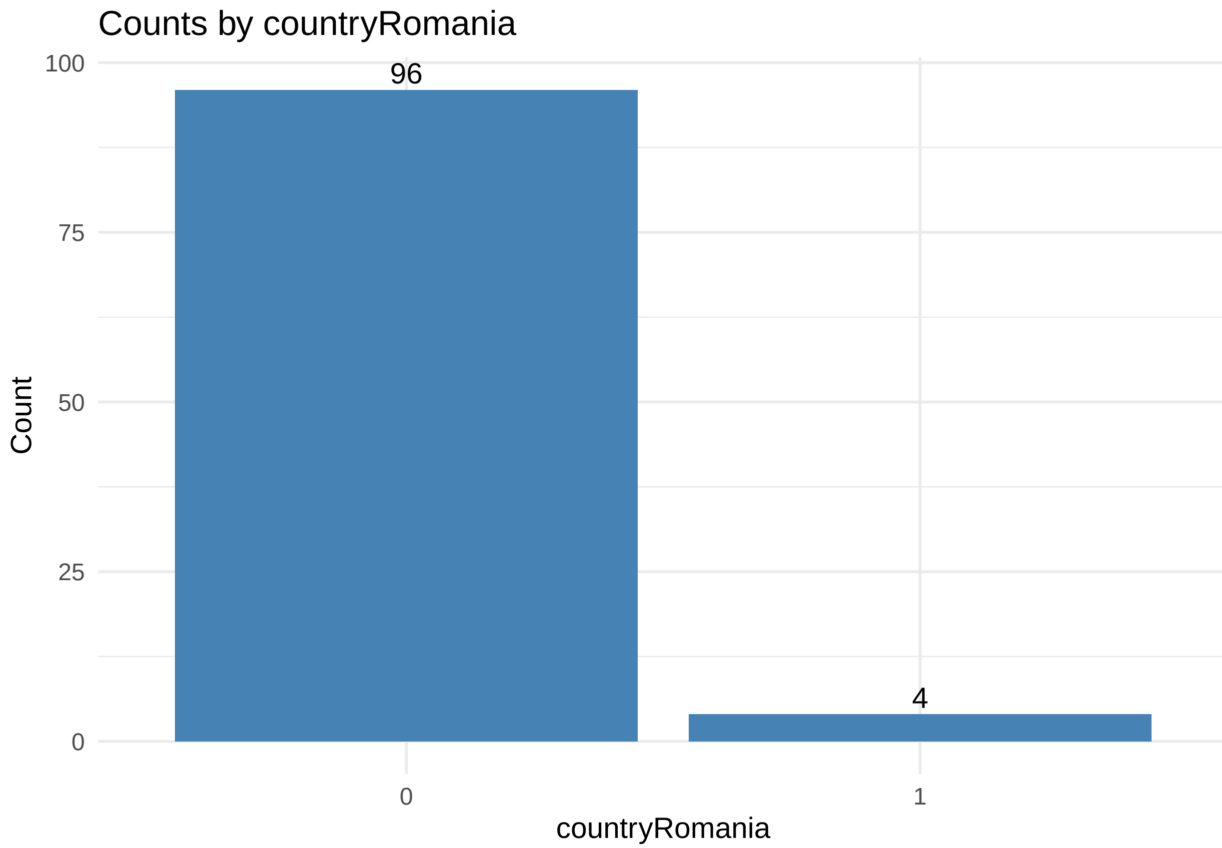

Counts by countryLatvia
99
1
100
75
50
25
0
Count
0
1
countryLatvia

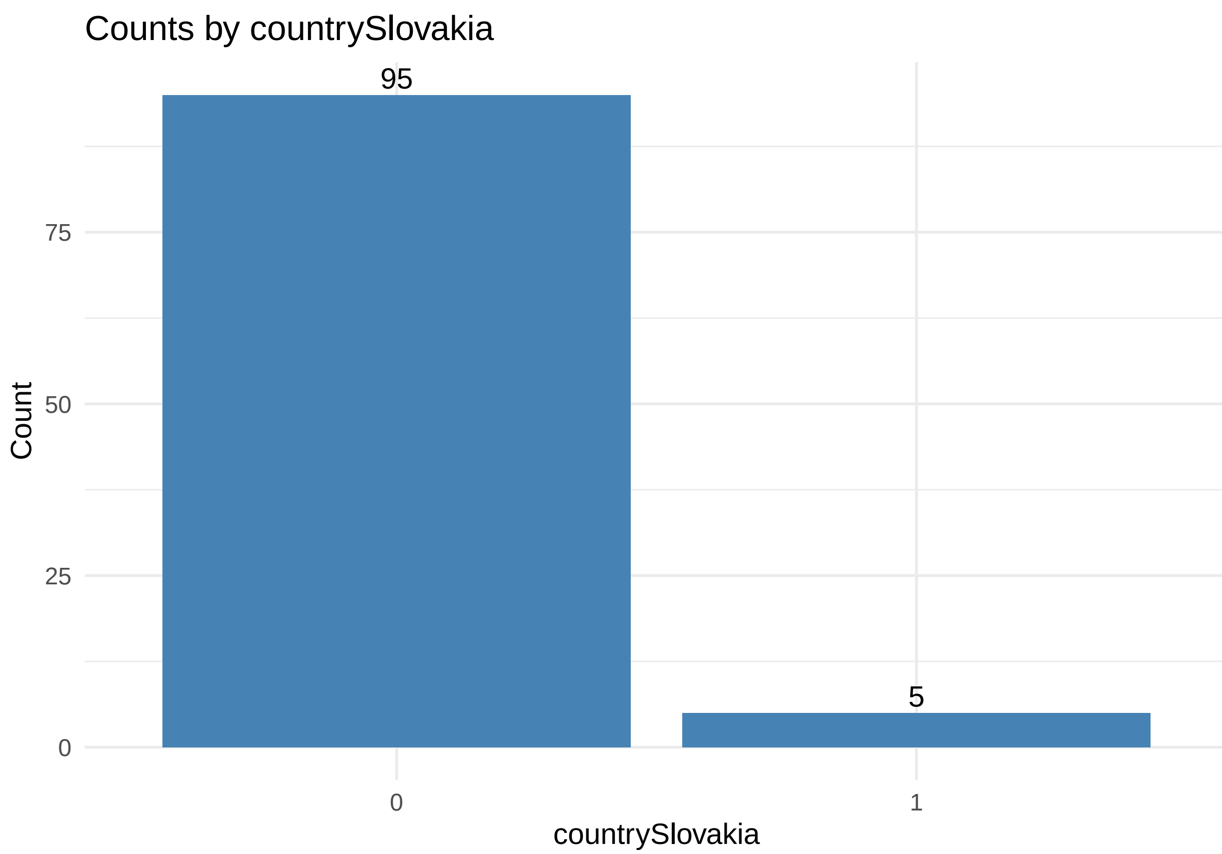

Counts by countryLuxembourg
100
75
50
25
0
Count
97
3
0
1
countryLuxembourg

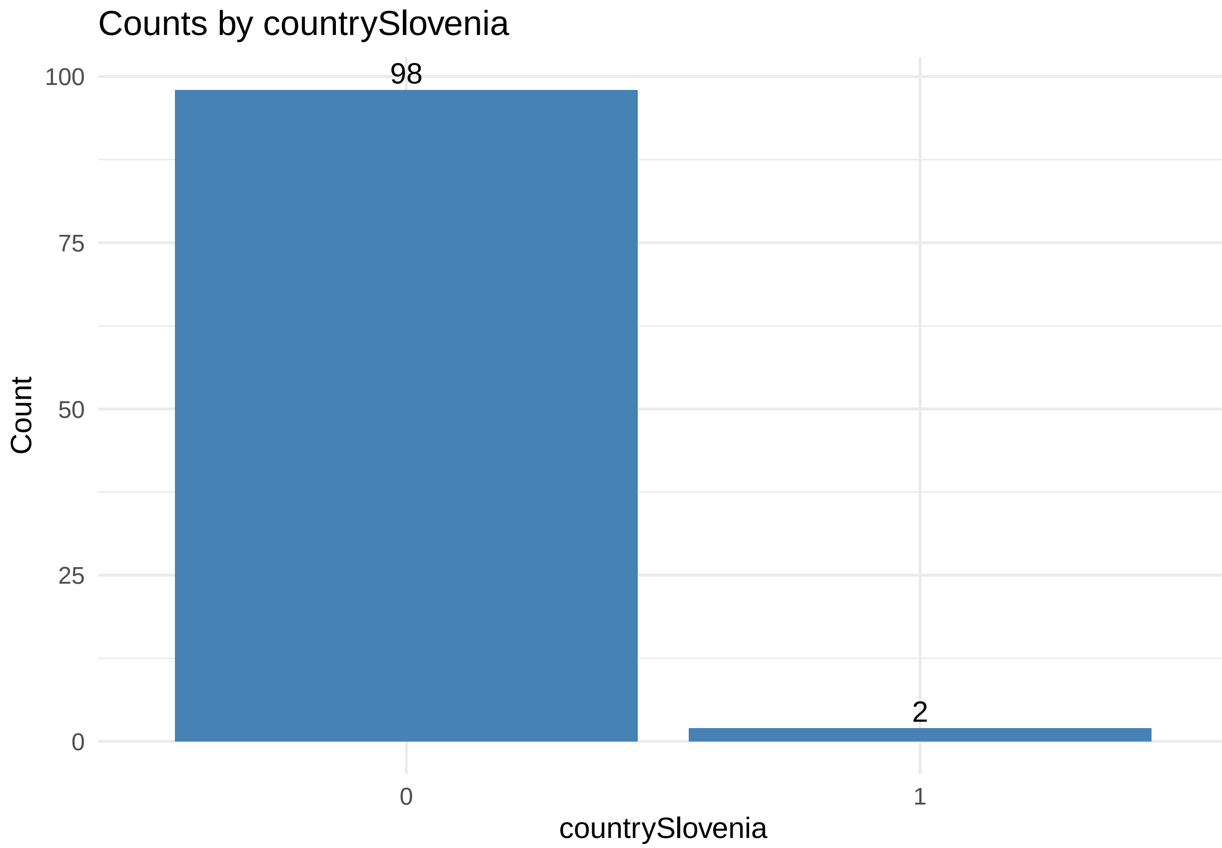

Counts by countryMalta
99
1
100
75
50
25
0
Count
0
1
countryMalta

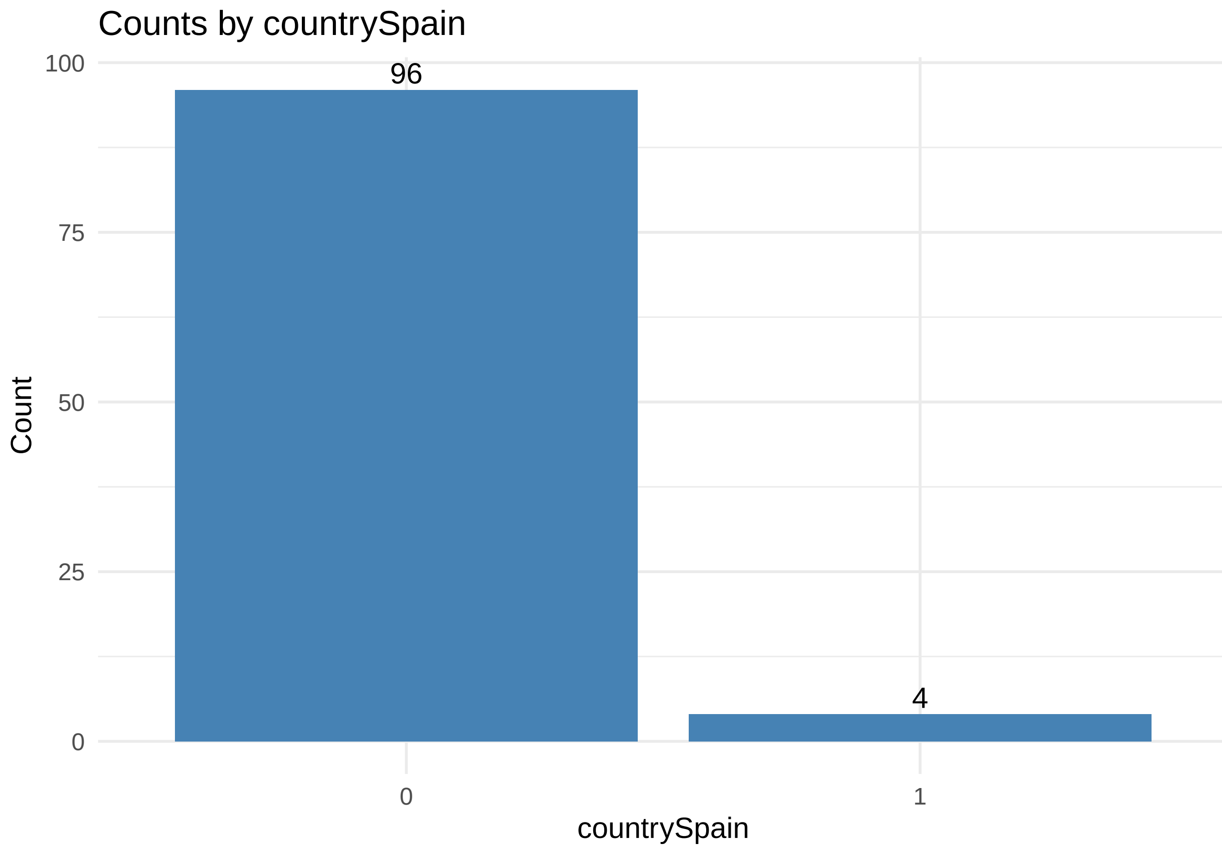

Counts by countryPoland
98
2
100
75
50
25
0
Count
0
1
countryPoland

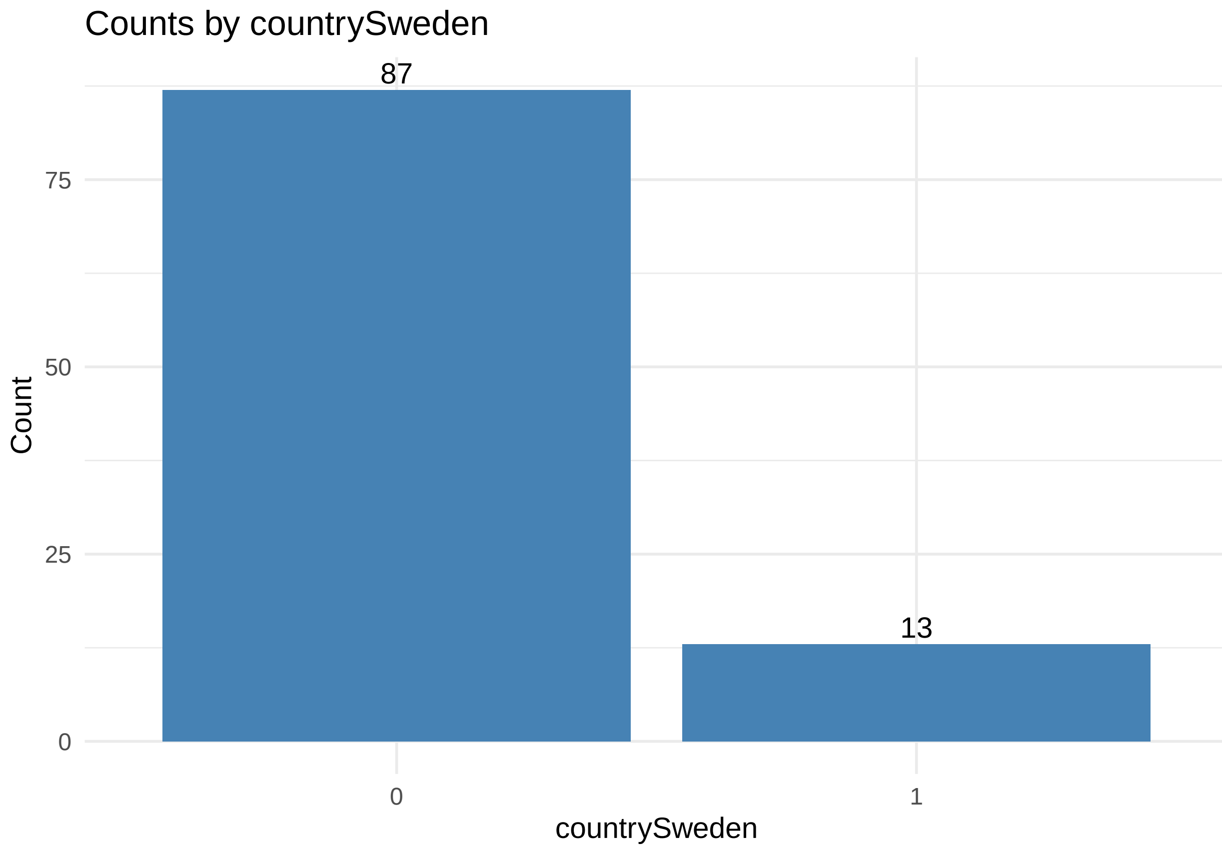

Counts by countryPortugal
99
1
100
75
50
25
0
Count
0
1
countryPortugal

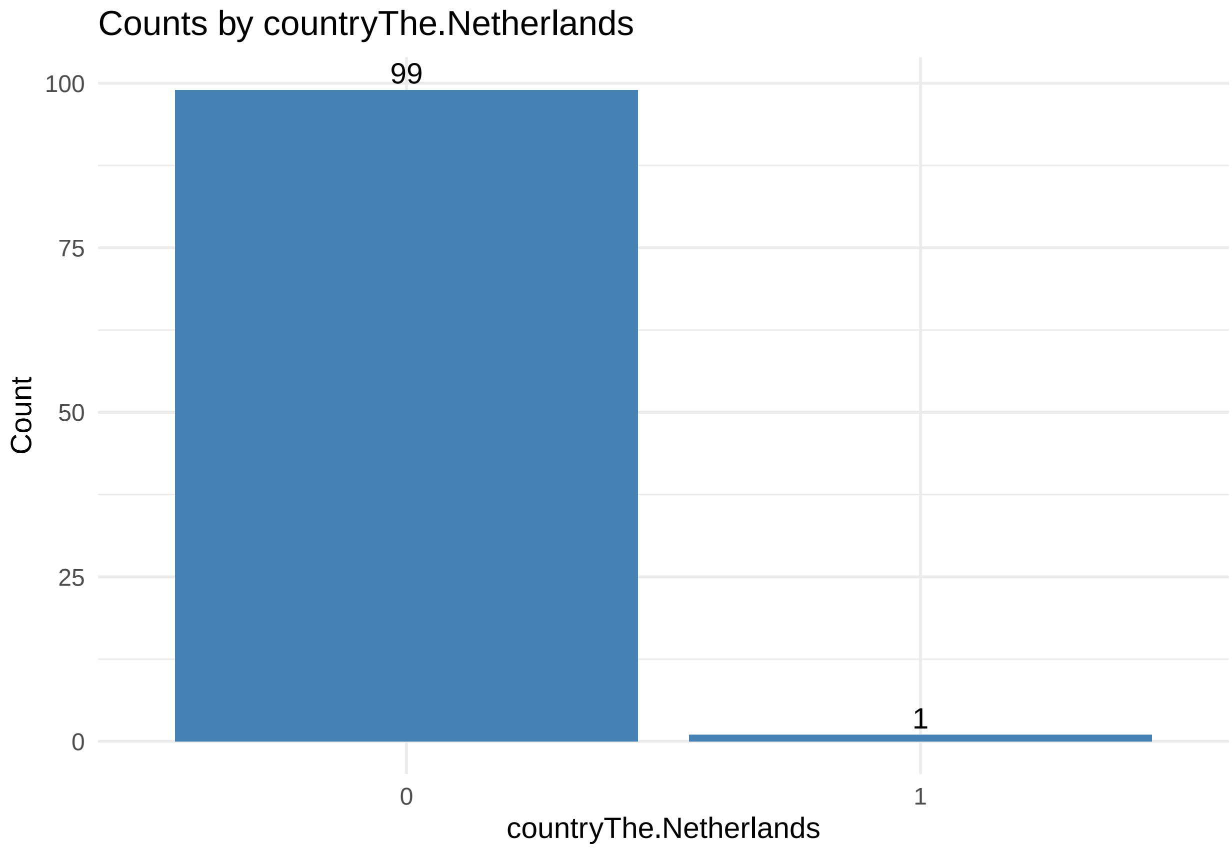

Counts by countryRomania
100
75
50
25
0
Count
96
4
0
1
countryRomania

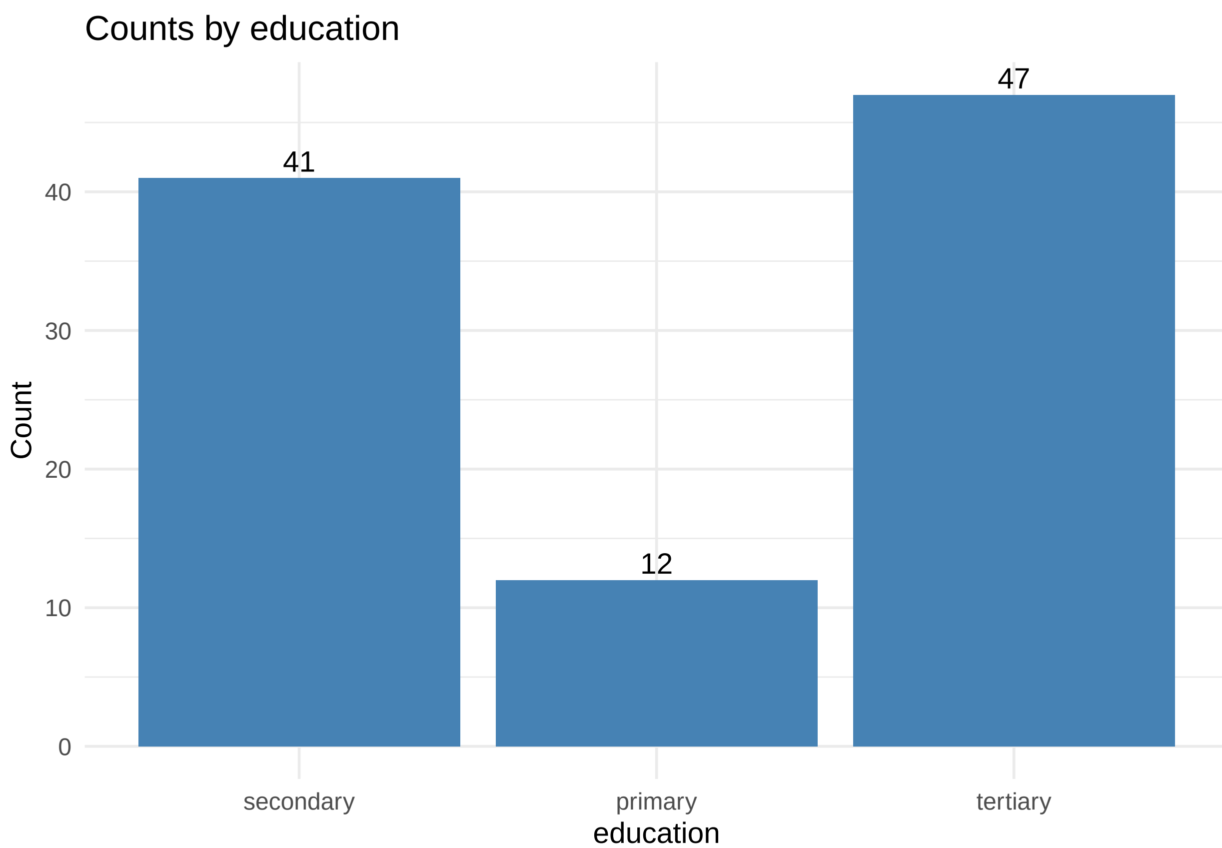

Counts by countrySlovakia
95
5
Count
0
25
50
75
0
1
countrySlovakia

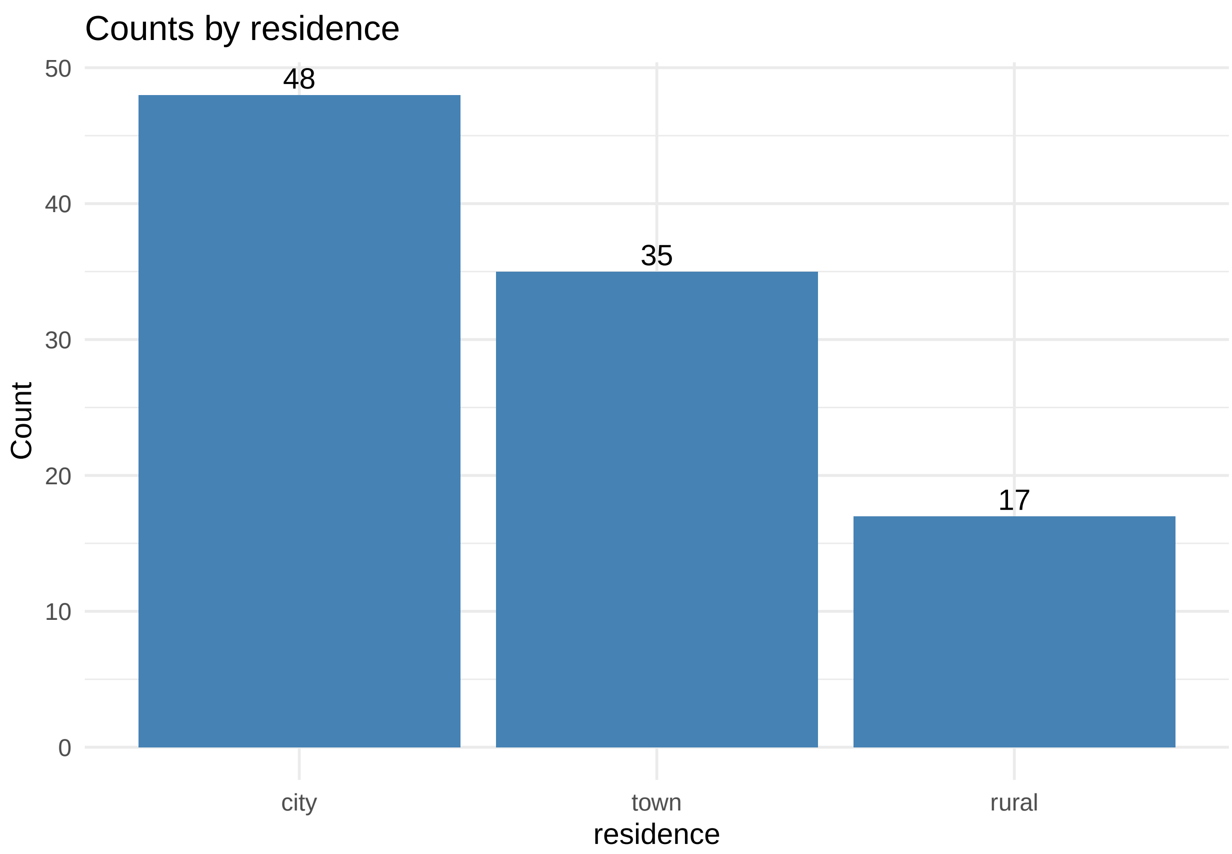

Counts by countrySlovenia
98
2
100
75
50
25
0
Count
0
1
countrySlovenia


Counts by countrySpain
100
75
50
25
0
Count
96
4
0
1
countrySpain


Counts by countrySweden
87
13
Count
75
50
25
0
0
1
countrySweden


Counts by countryThe.Netherlands
99
1
100
75
50
25
0
Count
0
1
countryThe.Netherlands


Counts by education
47
41
12
40
30
20
10
0
Count
secondary
primary
tertiary
education



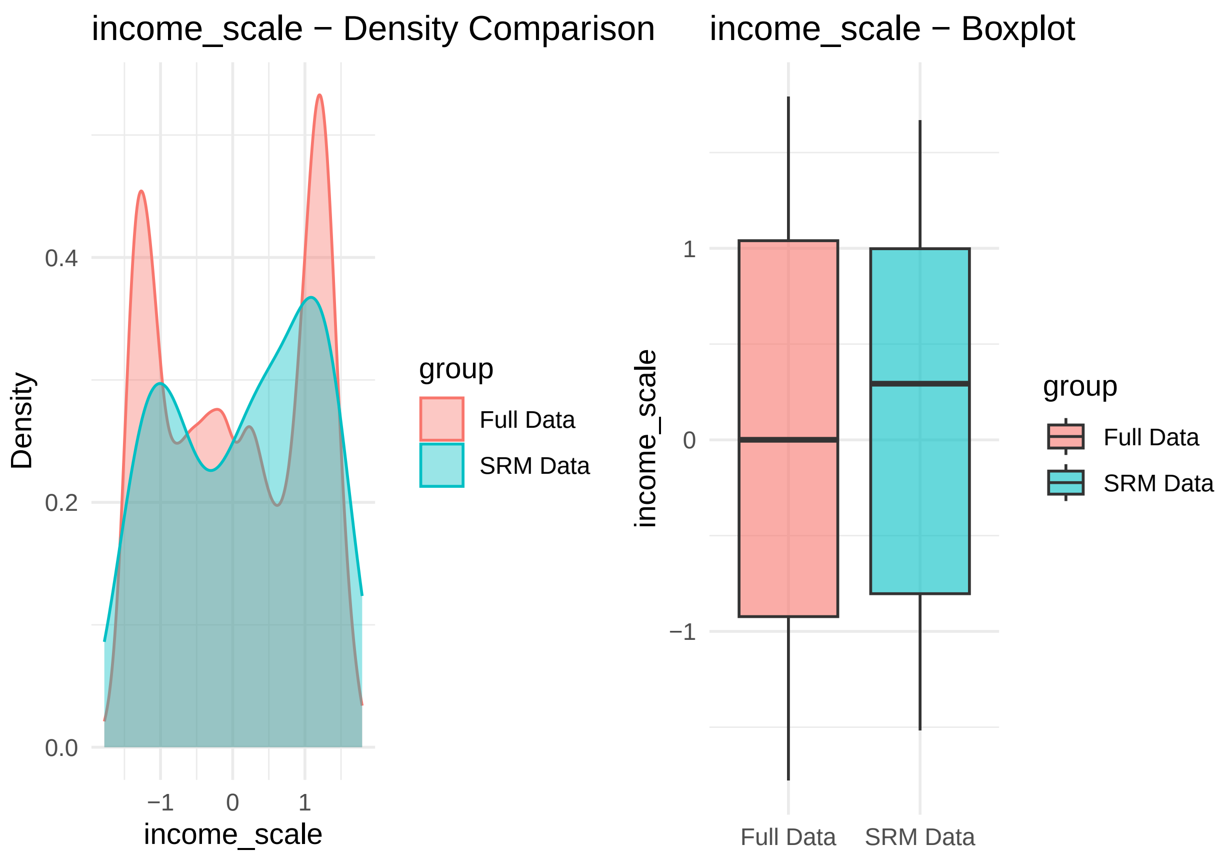

```
library(ggplot2)
library(dplyr)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:randomForest':
##
##     combine
```

```
compare_model_features_plot <- function(subset_data, full_data, model_formula) {
  model_vars <- all.vars(model_formula)[-1]  # exclude response variable

  for (var in model_vars) {
    # Skip if variable is missing in either dataset
    if (!(var %in% names(subset_data)) || !(var %in% names(full_data))) next

    x_subset <- subset_data[[var]]
    x_full   <- full_data[[var]]
```

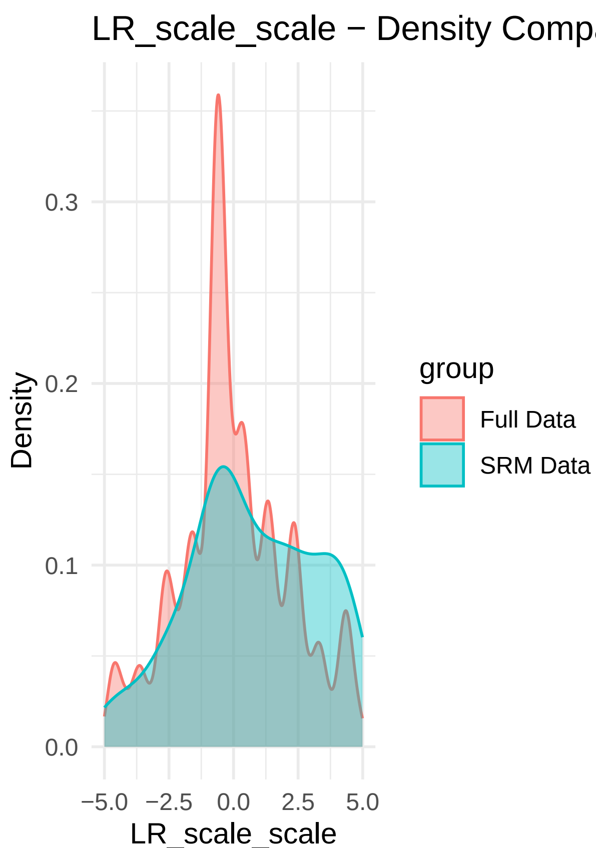
```r
    # Strict binary check (only 0 and 1, numeric)
    unique_vals <- unique(na.omit(c(x_subset, x_full)))
    is_binary <- is.numeric(x_subset) && length(unique_vals) == 2 && all(unique_vals %in% c(0, 1))
    is_categorical <- is.factor(x_subset) || is.character(x_subset) || is_binary

    if (is_categorical) {
      # Treat all as factors (including binary)
      all_levels <- union(unique(as.character(x_subset)), unique(as.character(x_full)))
      subset_data[[var]] <- factor(as.character(x_subset), levels = all_levels)
      full_data[[var]]   <- factor(as.character(x_full),   levels = all_levels)

      p1 <- ggplot(subset_data, aes_string(x = var)) +
        geom_bar(fill = "skyblue") +
        labs(title = paste(var, "(SRM Data)"), x = NULL, y = "Count") +
        theme_minimal() +
        theme(axis.text.x = element_text(angle = 45, hjust = 1))

      p2 <- ggplot(full_data, aes_string(x = var)) +
        geom_bar(fill = "salmon") +
        labs(title = paste(var, "(Full Data)"), x = NULL, y = "Count") +
        theme_minimal() +
        theme(axis.text.x = element_text(angle = 45, hjust = 1))

    } else if (is.numeric(x_subset)) {
      # Continuous numeric variable
      df_subset <- data.frame(value = x_subset, group = "SRM Data")
      df_full   <- data.frame(value = x_full, group = "Full Data")
      combined  <- bind_rows(df_subset, df_full)

      p1 <- ggplot(combined, aes(x = value, fill = group, color = group)) +
        geom_density(alpha = 0.4) +
        labs(title = paste(var, "- Density Comparison"), x = var, y = "Density") +
        theme_minimal()

      p2 <- ggplot(combined, aes(x = group, y = value, fill = group)) +
        geom_boxplot(alpha = 0.6) +
        labs(title = paste(var, "- Boxplot"), x = NULL, y = var) +
        theme_minimal()
    } else {
      next
    }

    # Display side-by-side plots
    grid.arrange(p1, p2, ncol = 2)
  }
}
```

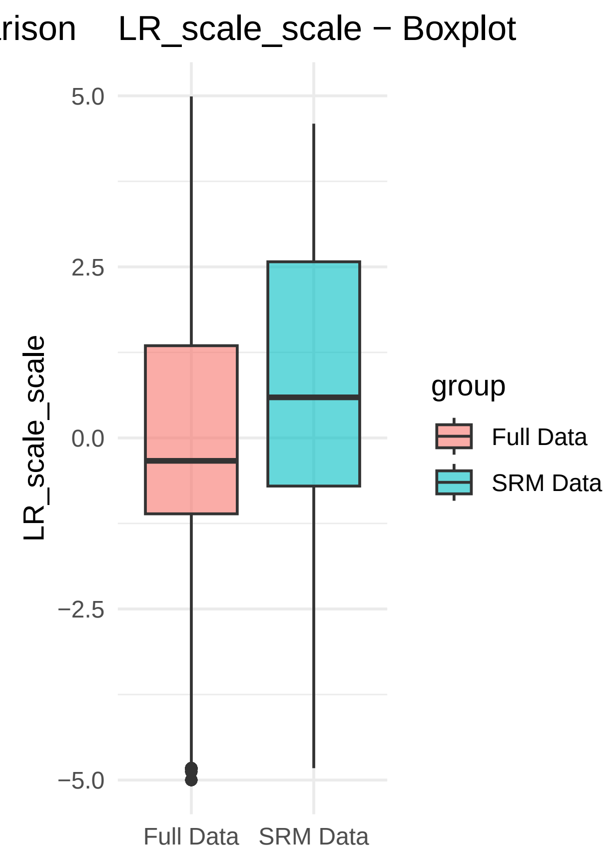
```r
compare_model_features_plot(srm_final, df_clean, model_formula = full_formula)
```

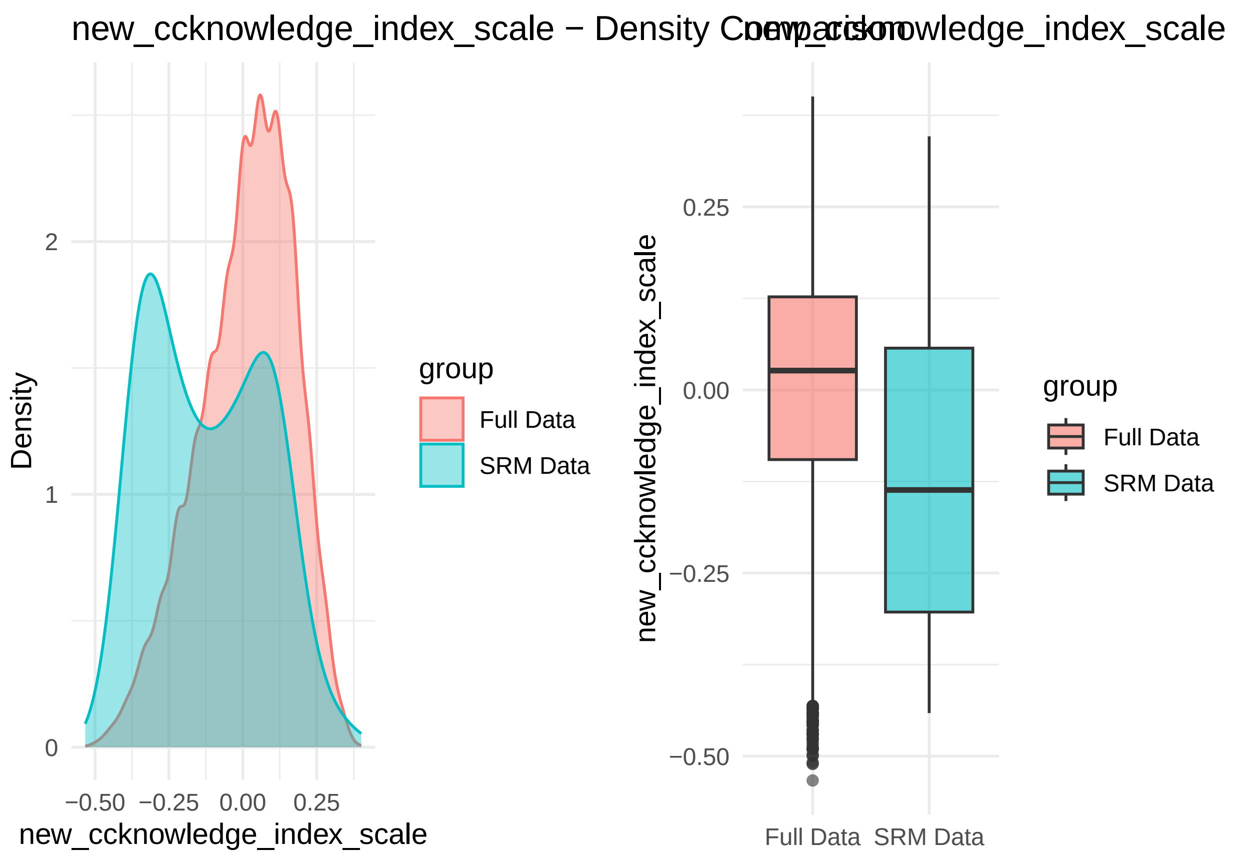


income_scale − Density Comparison
Density
0.4
0.2
0.0
−1
0
1
income_scale
group
Full Data
SRM Data
income_scale − Boxplot
income_scale
1
0
−1
Full Data
SRM Data
group
Full Data
SRM Data

trust (SRM Data)
Count
0
10
20
30
No really confident
Very confident
Rather confident
No confident at all
trust (Full Data)
Count
0
2000
4000
6000
8000
No really confident
Very confident
Rather confident
No confident at all

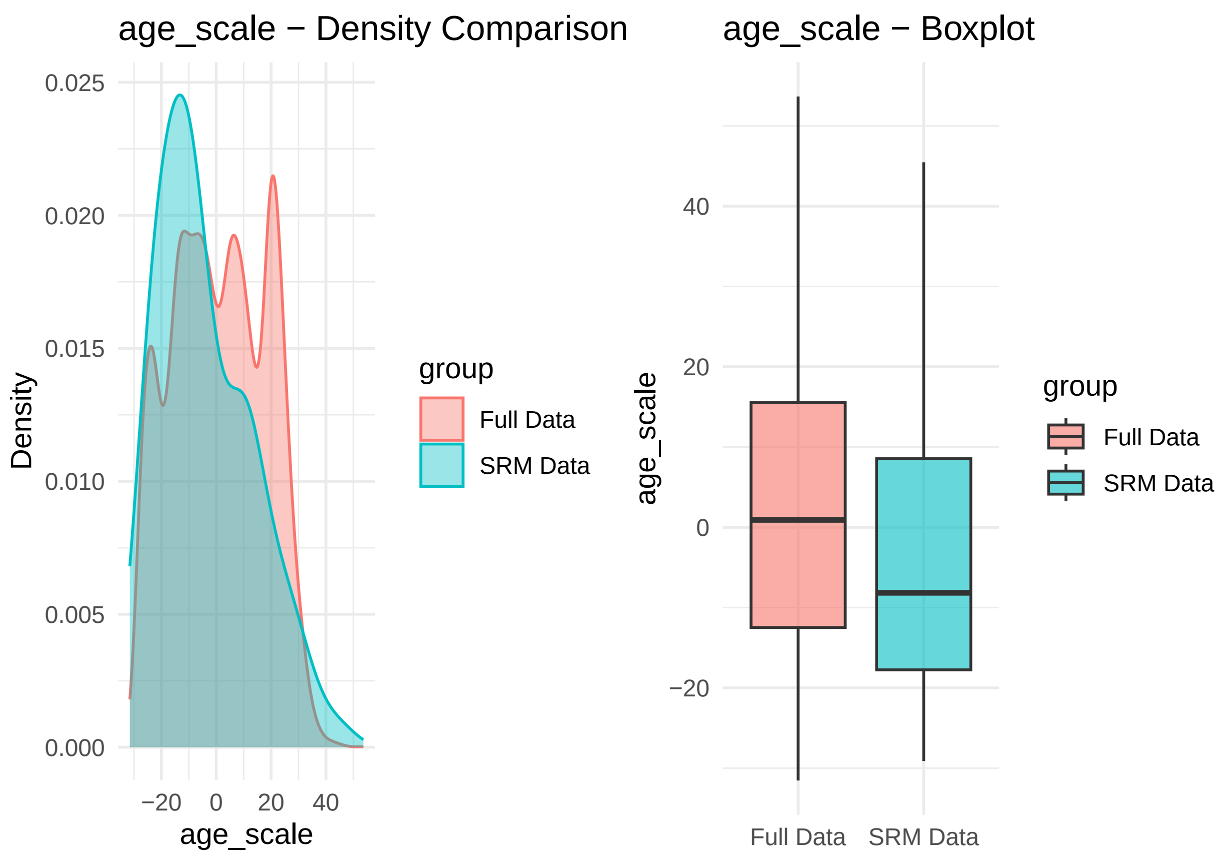

LR_scale_scale − Density Comparison
Density
0.0
0.1
0.2
0.3
−5.0
−2.5
0.0
2.5
5.0
LR_scale_scale
group
Full Data
SRM Data


LR_scale_scale − Boxplot
LR_scale_scale
5.0
2.5
0.0
−2.5
−5.0
Full Data
SRM Data
group
Full Data
SRM Data

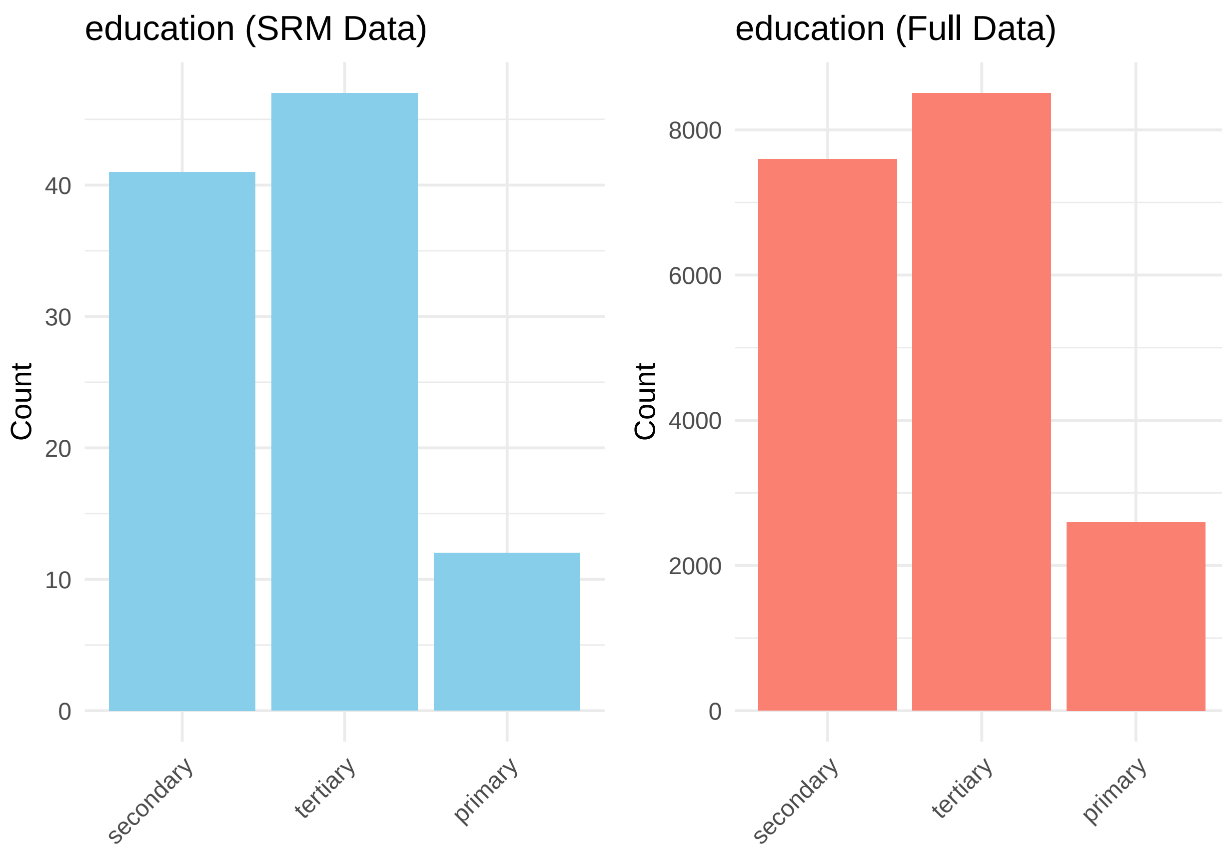


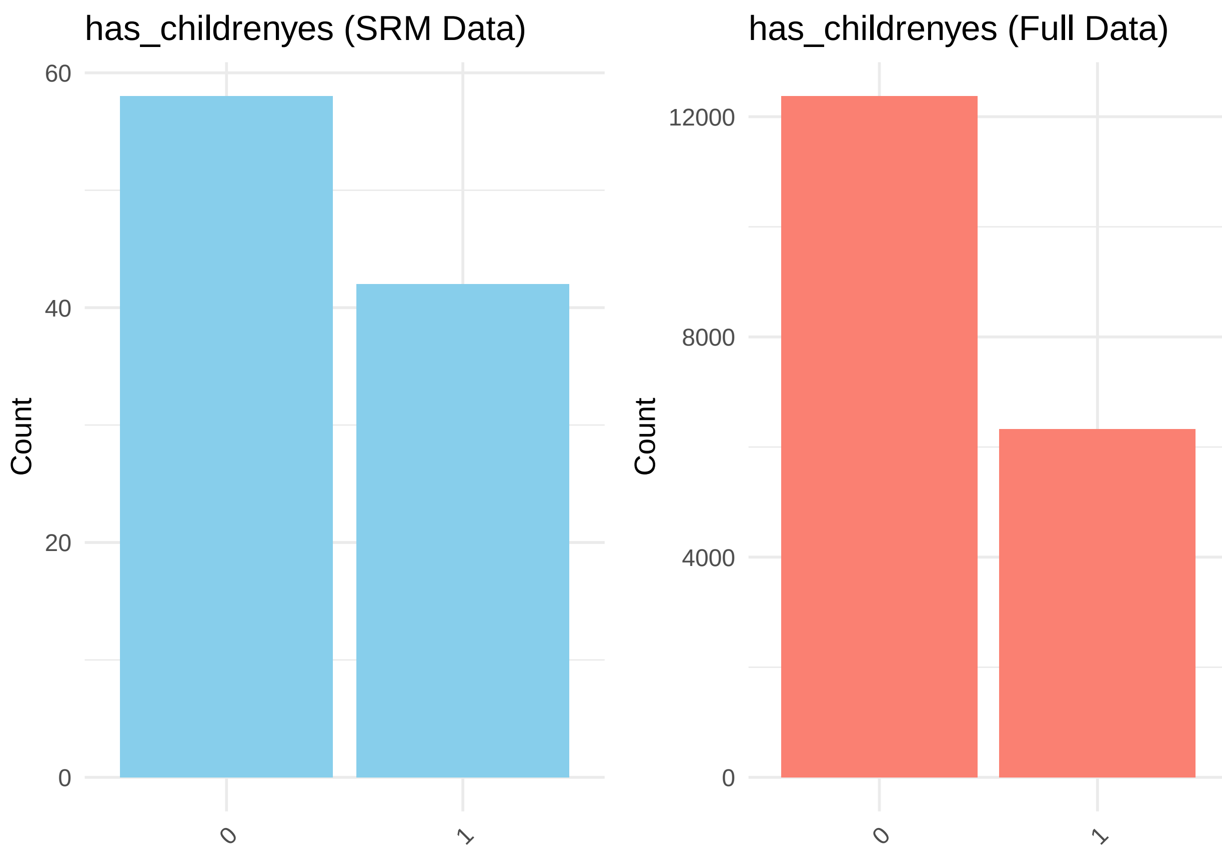
new_ccknowledge_index_scale − Density Comparison
new_ccknowledge_index_scale -
Density
2
1
0
−0.50
−0.25
0.00
0.25
new_ccknowledge_index_scale
group
Full Data
SRM Data
new_ccknowledge_index_scale
0.25
0.00
−0.25
−0.50
Full Data
SRM Data
group
Full Data
SRM Data

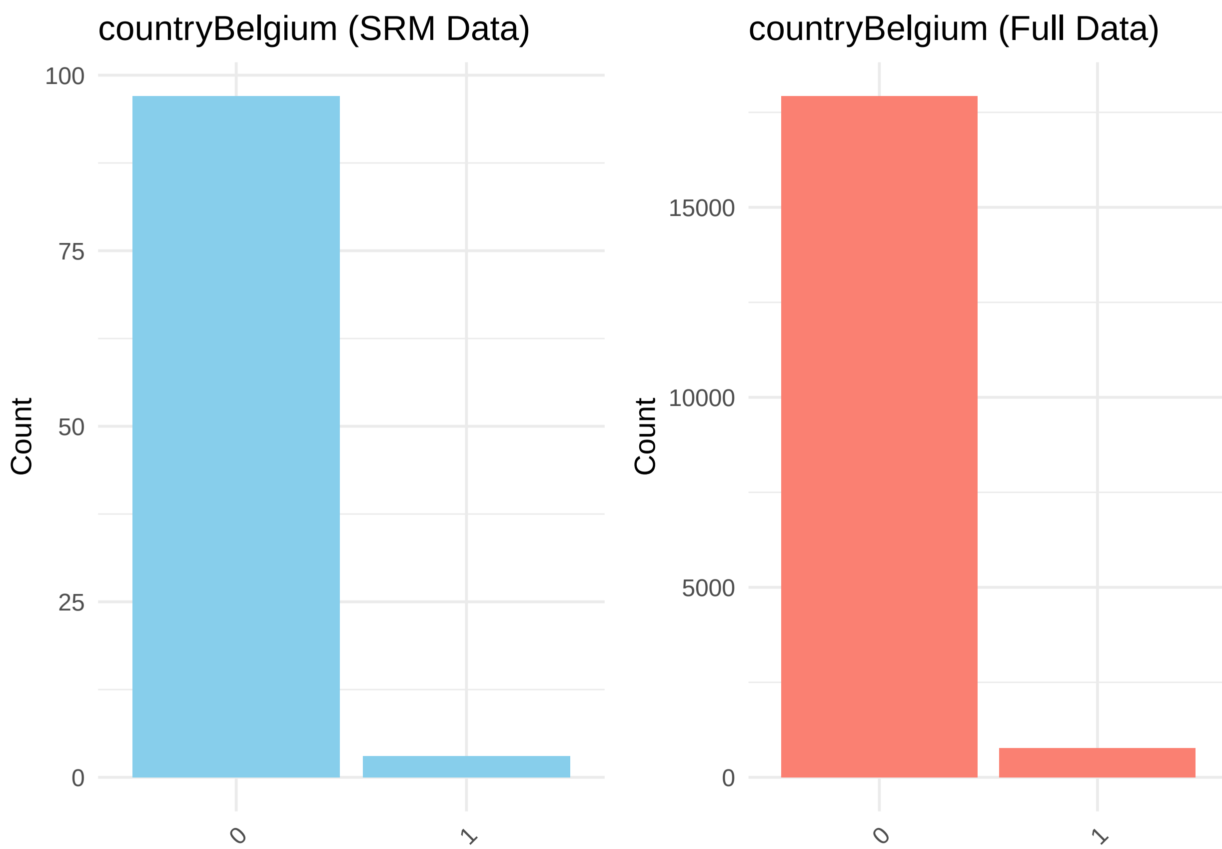

residence (SRM Data)
Count
50
40
30
20
10
0
town
rural
city
residence (Full Data)
Count
8000
6000
4000
2000
0
town
rural
city

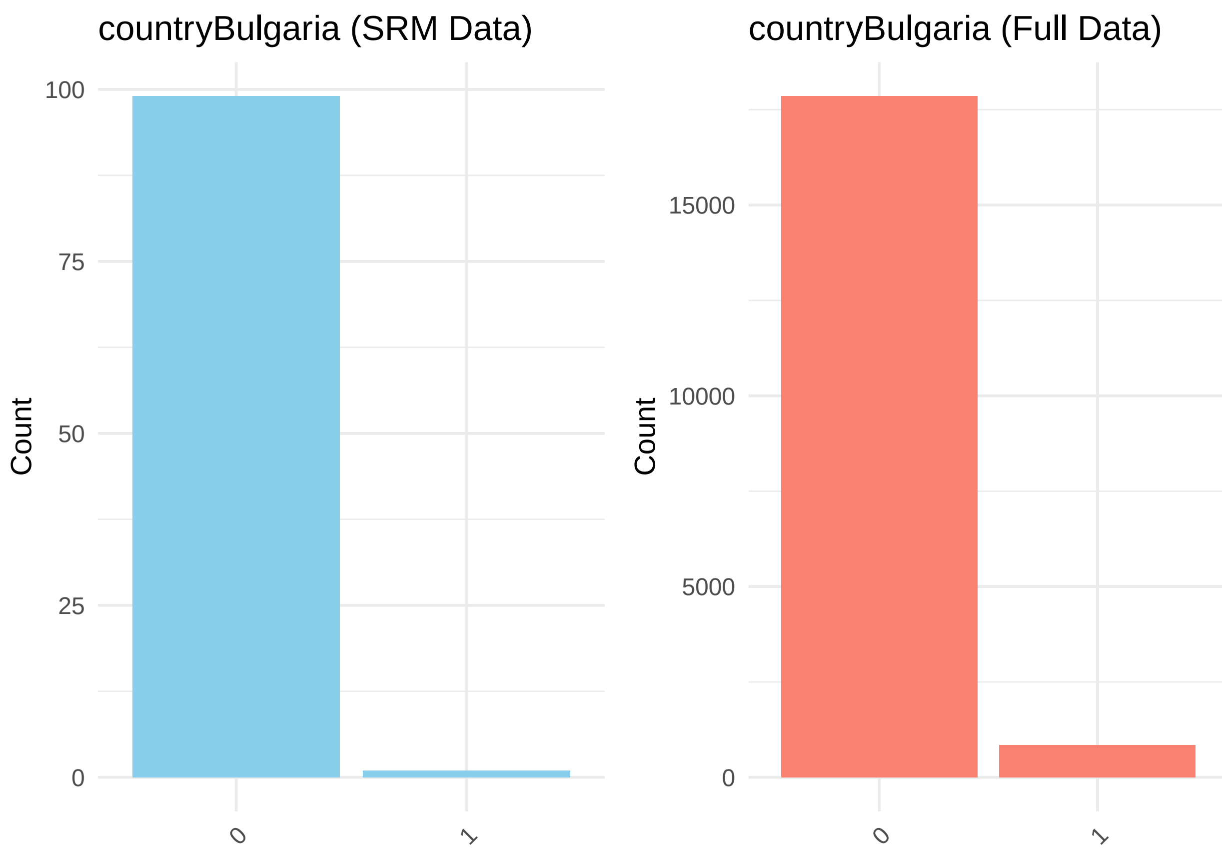

age_scale − Density Comparison
0.025
0.020
0.015
0.010
0.005
0.000
Density
−20
0
20
40
age_scale
group
Full Data
SRM Data
age_scale − Boxplot
40
20
0
−20
age_scale
Full Data
SRM Data
group
Full Data
SRM Data


gender (SRM Data)
Count
60
40
20
0
male
female
gender (Full Data)
Count
7500
5000
2500
0
male
female

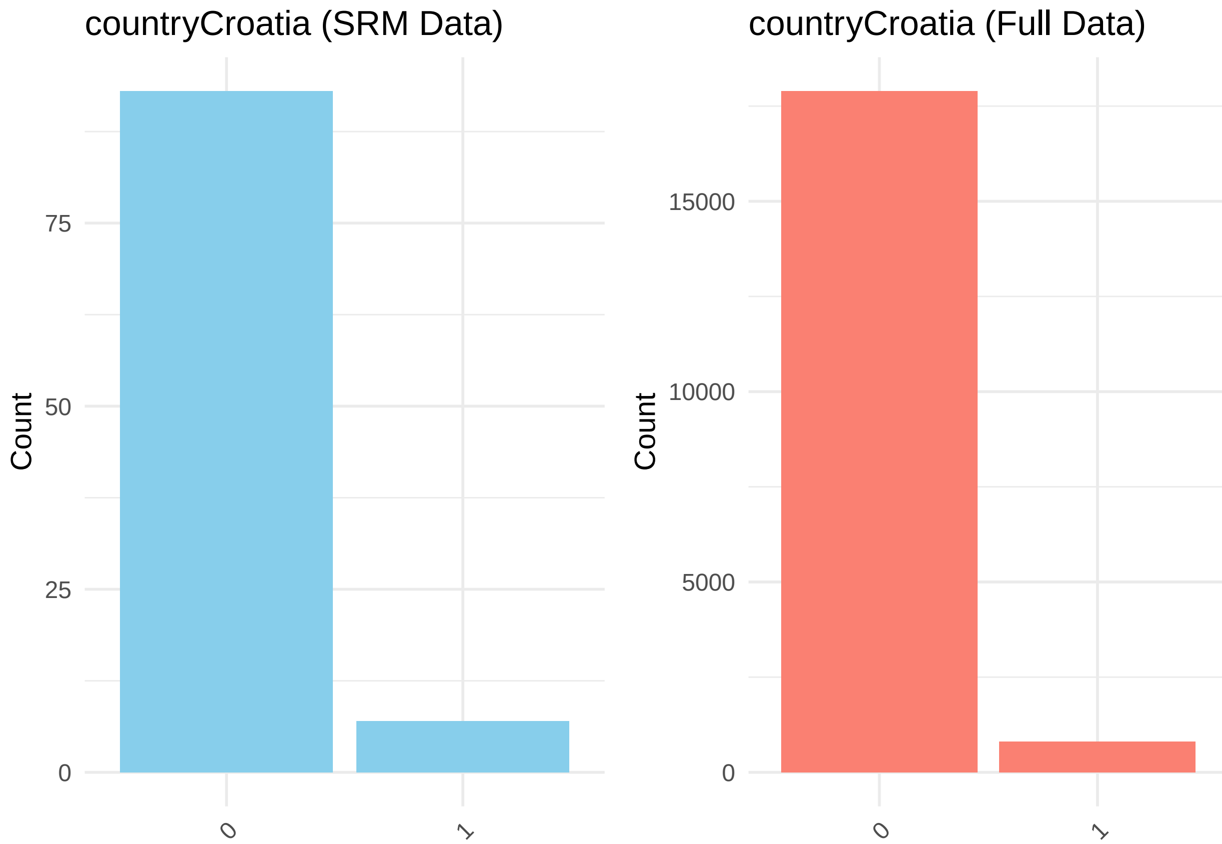

education (SRM Data)
Count
0
10
20
30
40
secondary
tertiary
primary
education (Full Data)
Count
0
2000
4000
6000
8000
secondary
tertiary
primary

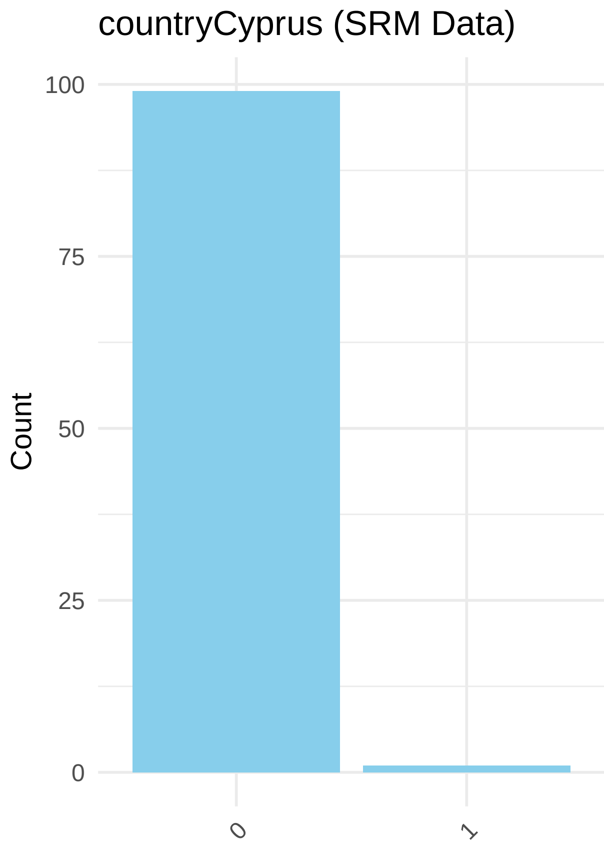

has_childrenyes (SRM Data)
Count
60
40
20
0
0
1
has_childrenyes (Full Data)
Count
12000
8000
4000
0
0
1

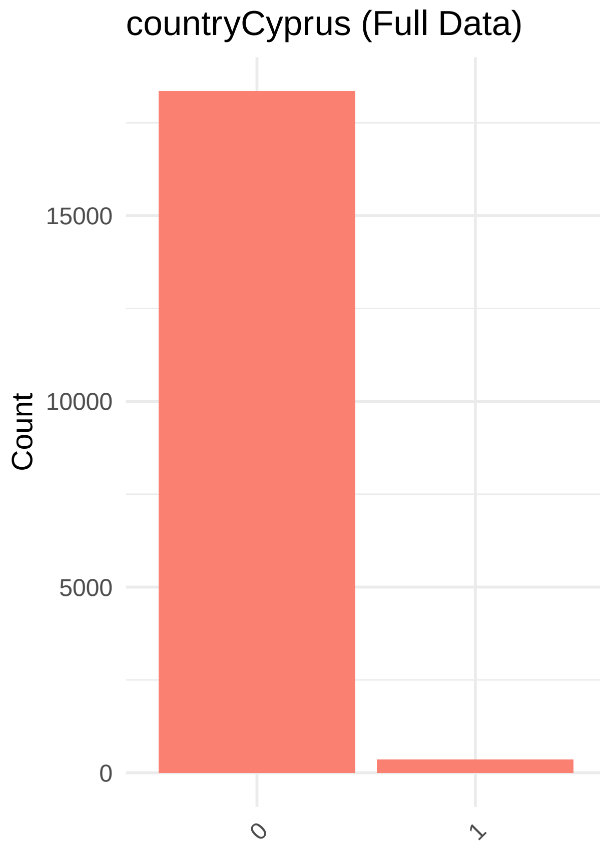

countryBelgium (SRM Data)
Count
100
75
50
25
0
0
1
countryBelgium (Full Data)
Count
15000
10000
5000
0
0
1

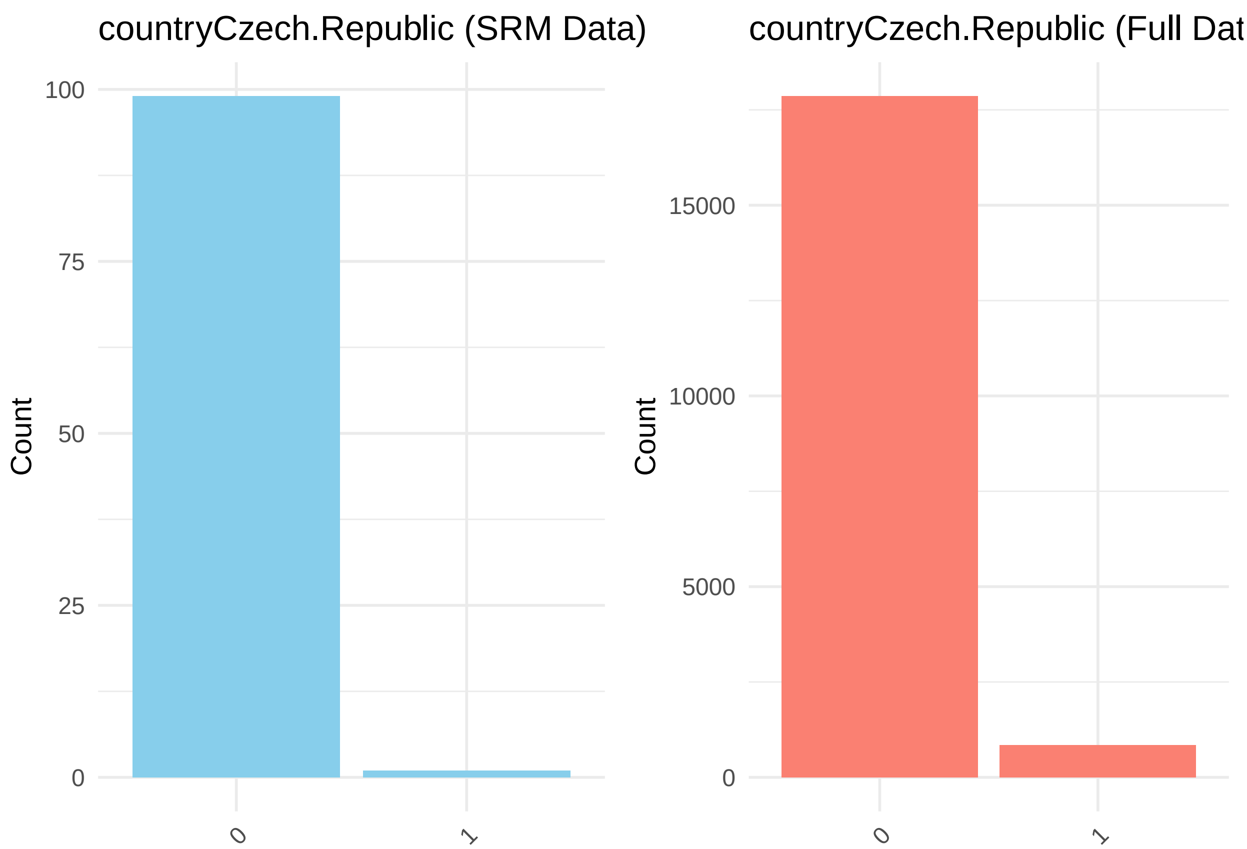


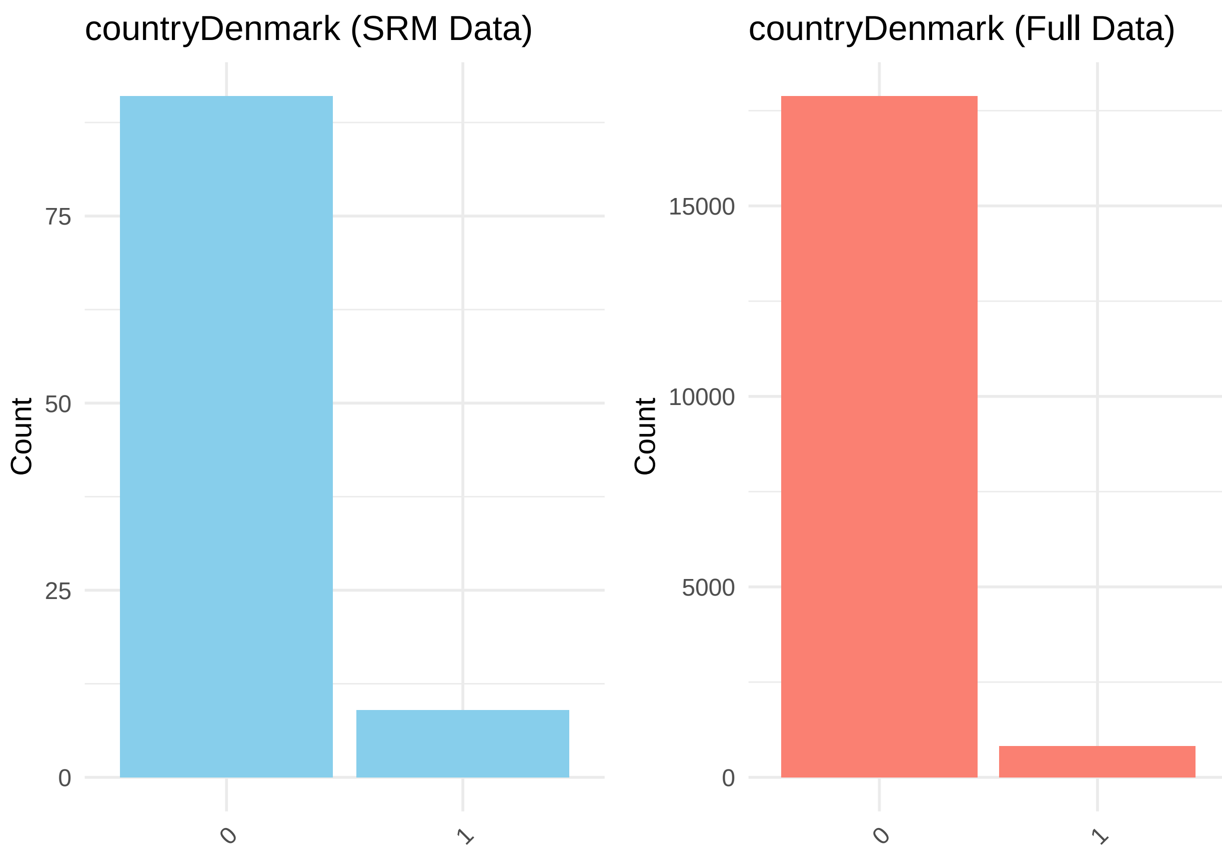
countryBulgaria (SRM Data)
Count
100
75
50
25
0
0
1
countryBulgaria (Full Data)
Count
15000
10000
5000
0
0
1

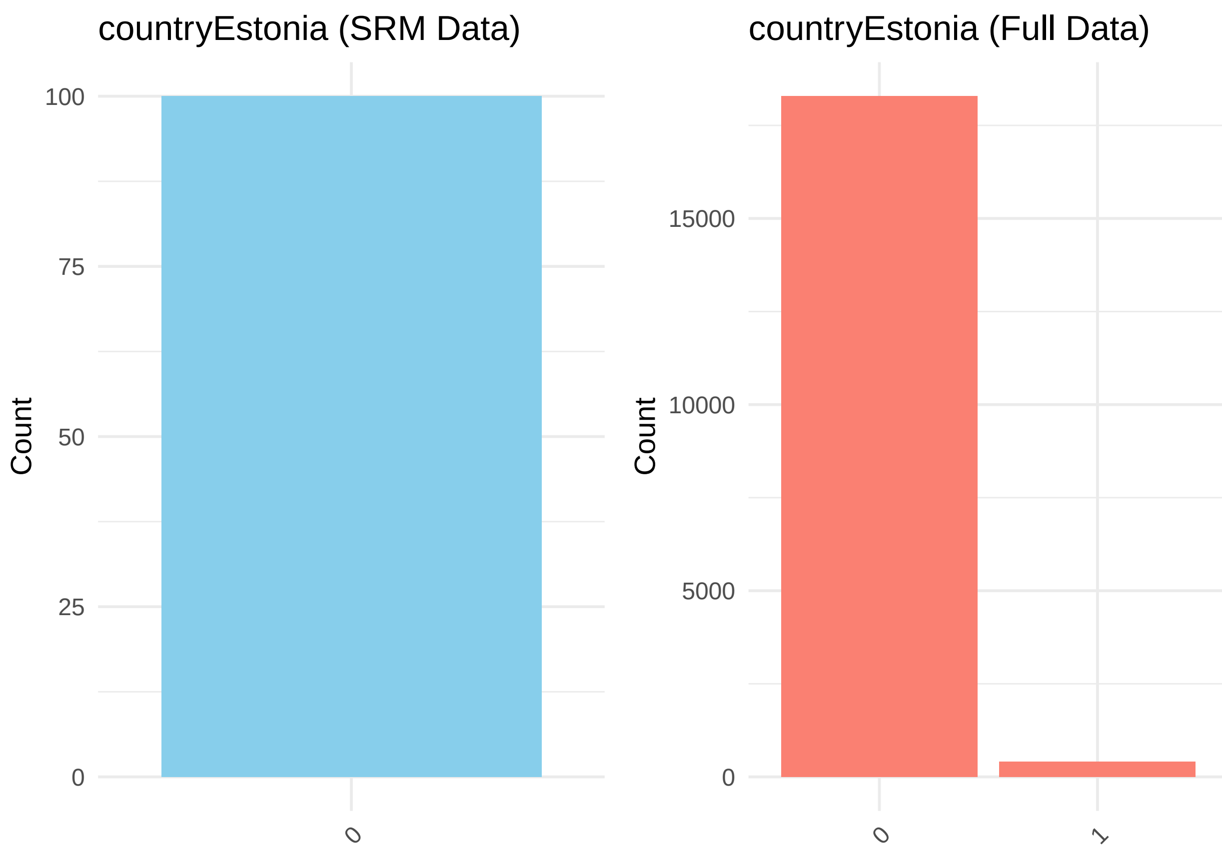

countryCroatia (SRM Data)
Count
0
25
50
75
0
1
countryCroatia (Full Data)
Count
0
5000
10000
15000
0
1

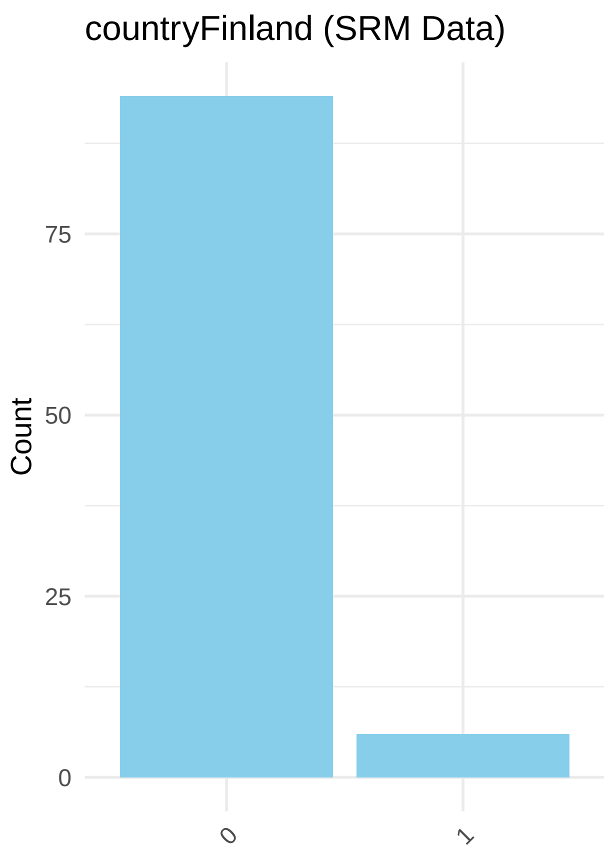

countryCyprus (SRM Data)
Count
100
75
50
25
0
0
1

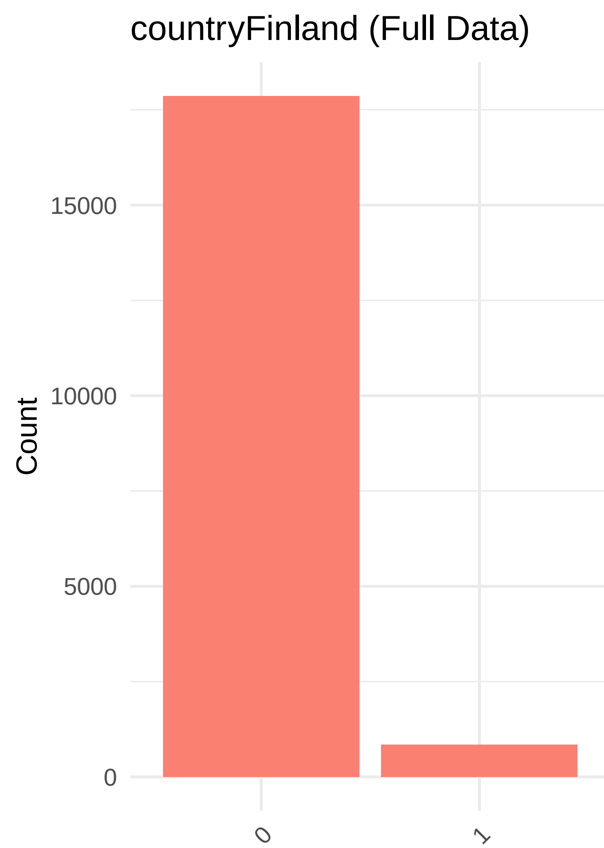

countryCyprus (Full Data)
Count
15000
10000
5000
0
0
1

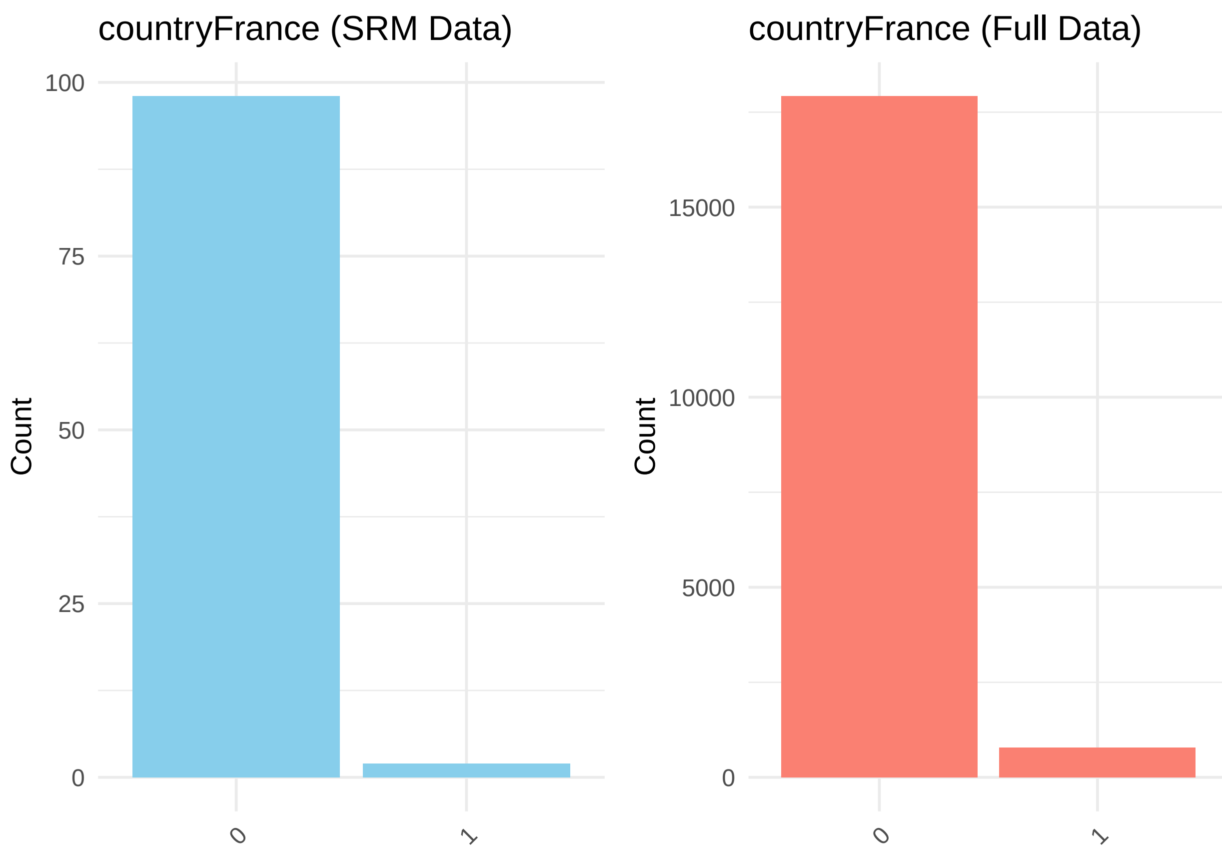

countryCzech.Republic (SRM Data)
Count
100
75
50
25
0
0
1
countryCzech.Republic (Full Dat
Count
15000
10000
5000
0
0
1

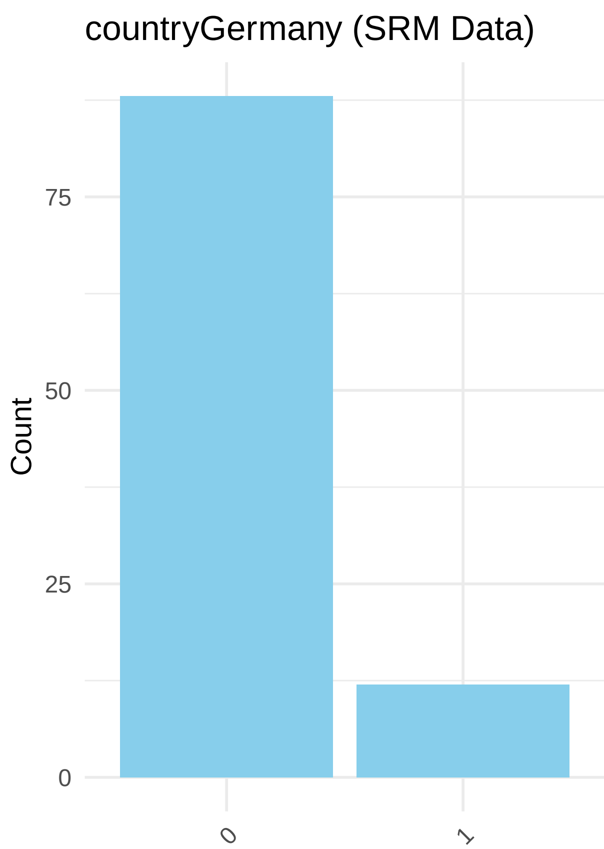

countryDenmark (SRM Data)
Count
0
25
50
75
0
1
countryDenmark (Full Data)
Count
0
5000
10000
15000
0
1

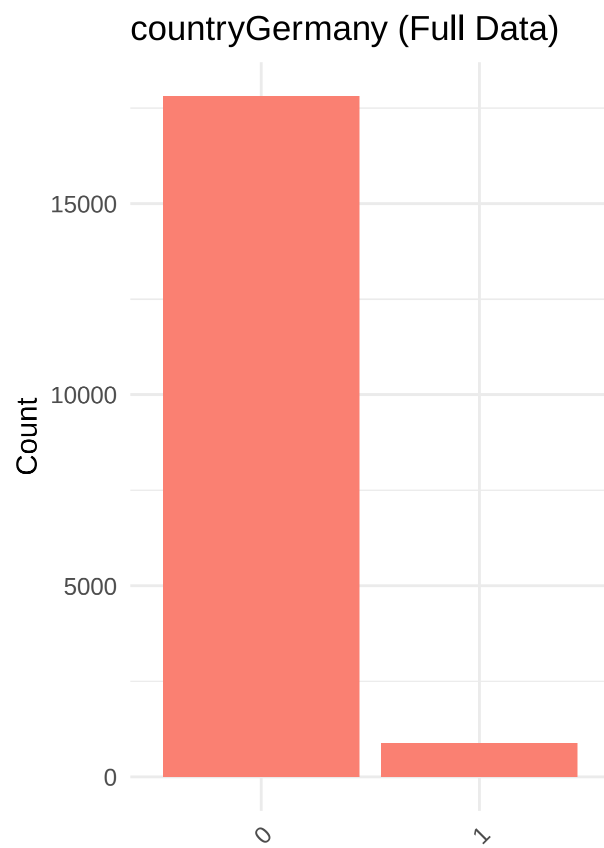

countryEstonia (SRM Data)
Count
100
75
50
25
0
0
countryEstonia (Full Data)
Count
15000
10000
5000
0
0
1

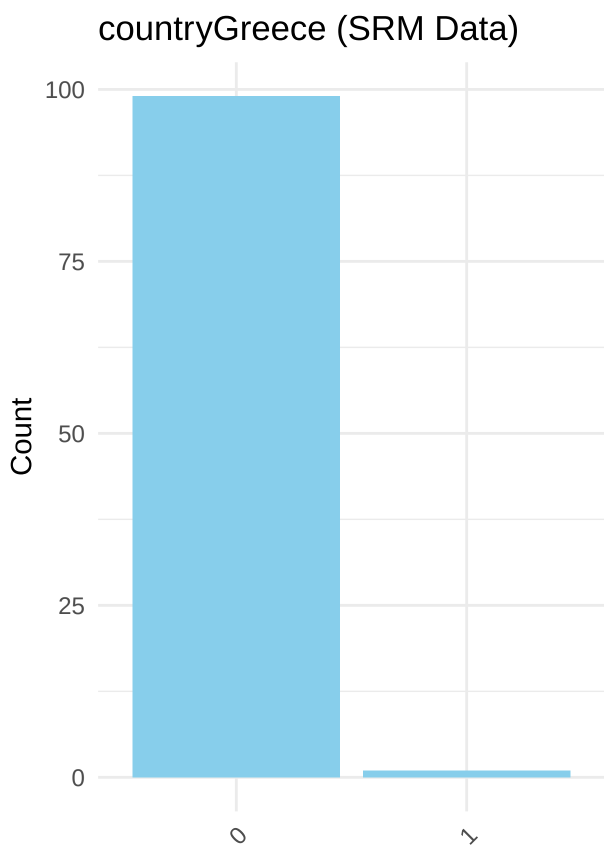

countryFinland (SRM Data)
Count
75
50
25
0
0
1


countryFinland (Full Data)
Count
15000
10000
5000
0
0
1

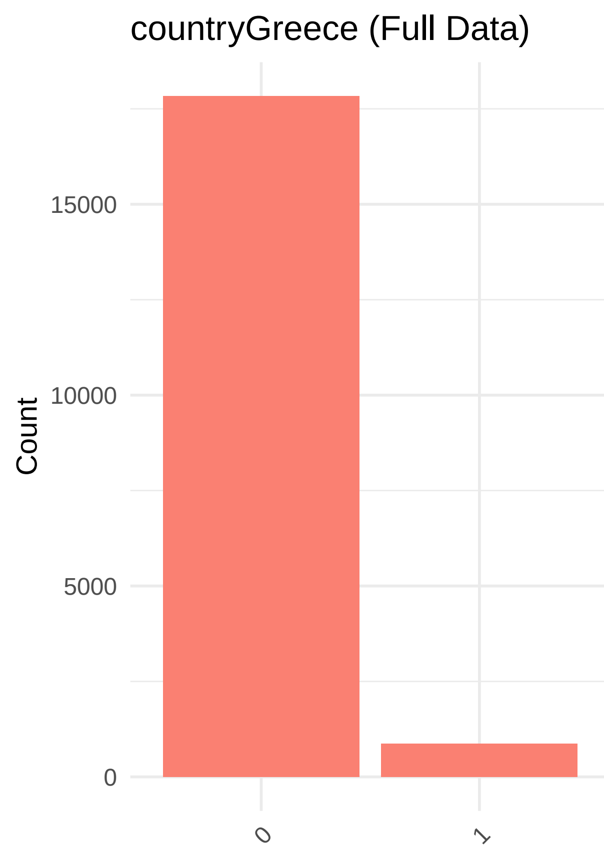

countryFrance (SRM Data)
Count
100
75
50
25
0
0
1
countryFrance (Full Data)
Count
15000
10000
5000
0
0
1


countryGermany (SRM Data)
Count
75
50
25
0
0
1

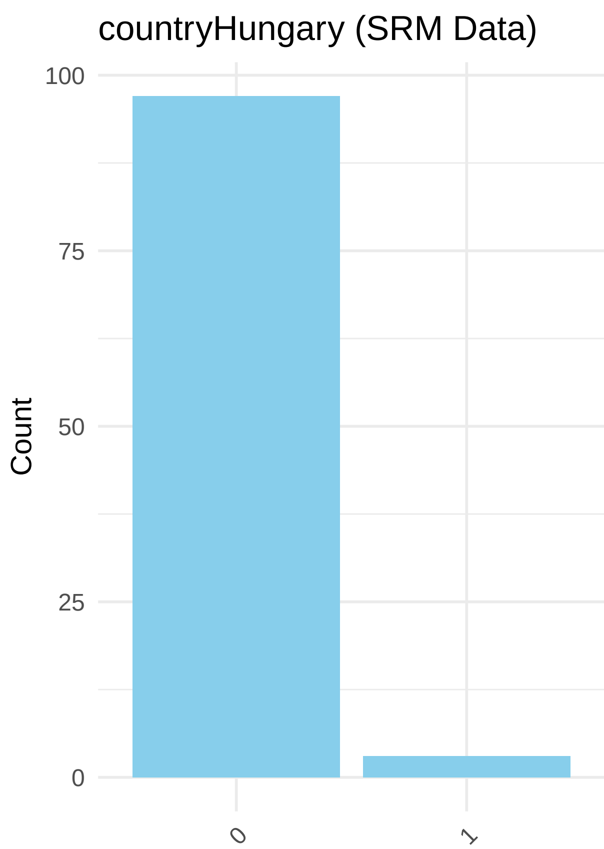

countryGermany (Full Data)
Count
15000
10000
5000
0
0
1

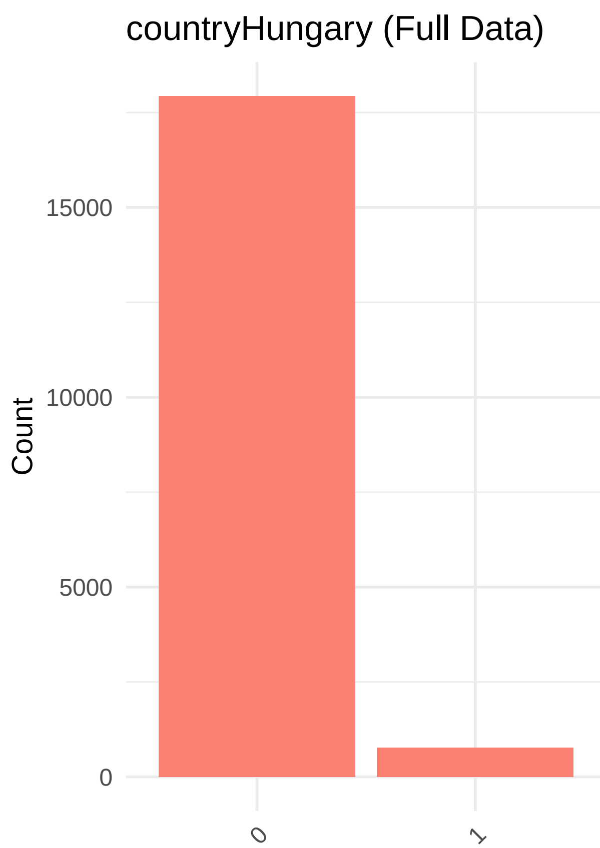


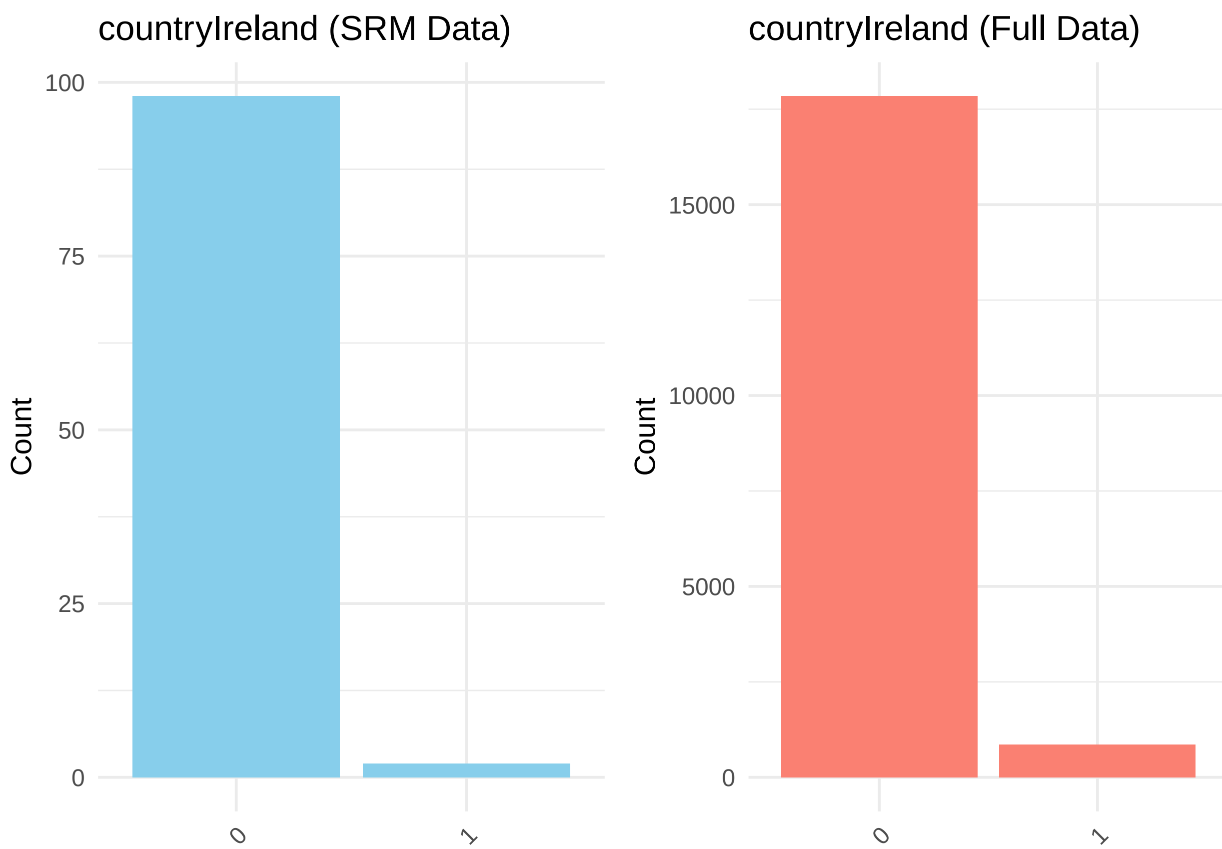
countryGreece (SRM Data)
Count
100
75
50
25
0
0
1



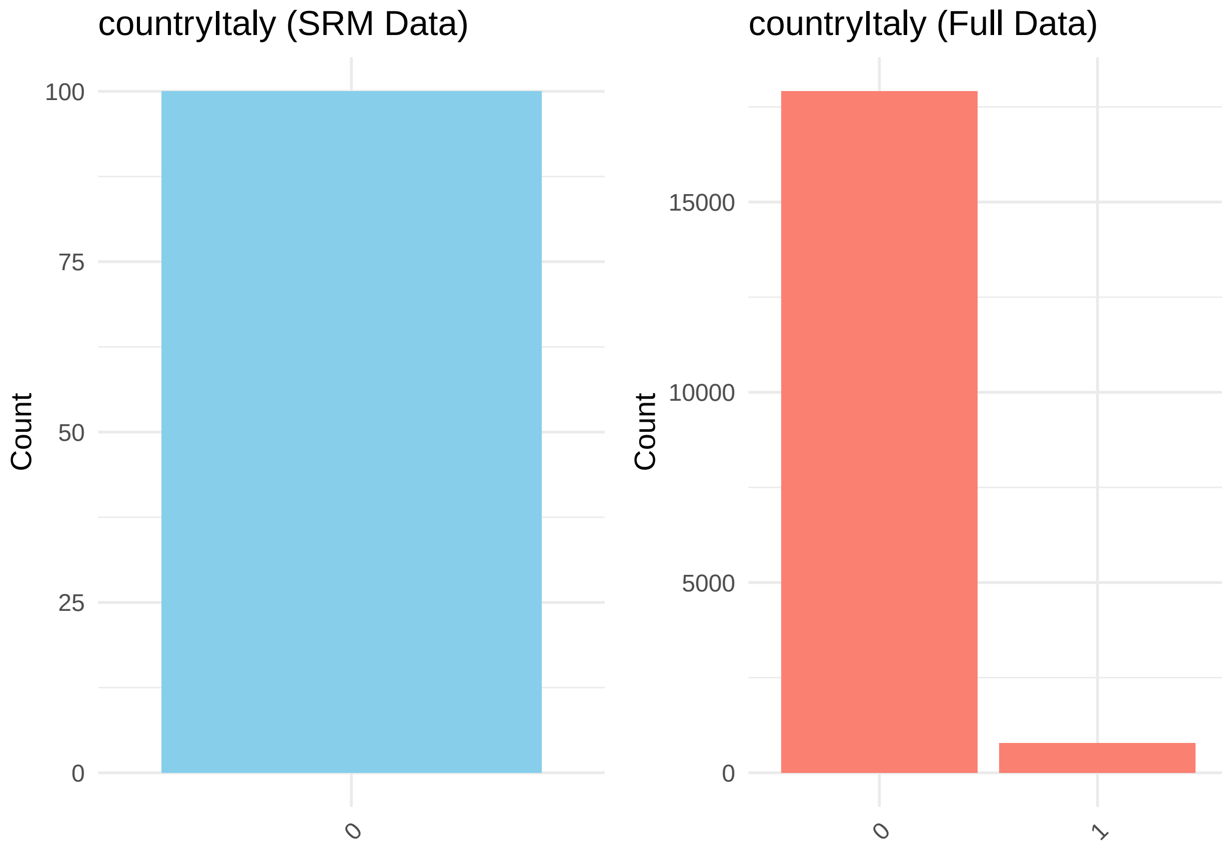
countryGreece (Full Data)
Count
15000
10000
5000
0
0
1


countryHungary (SRM Data)
Count
100
75
50
25
0
0
1

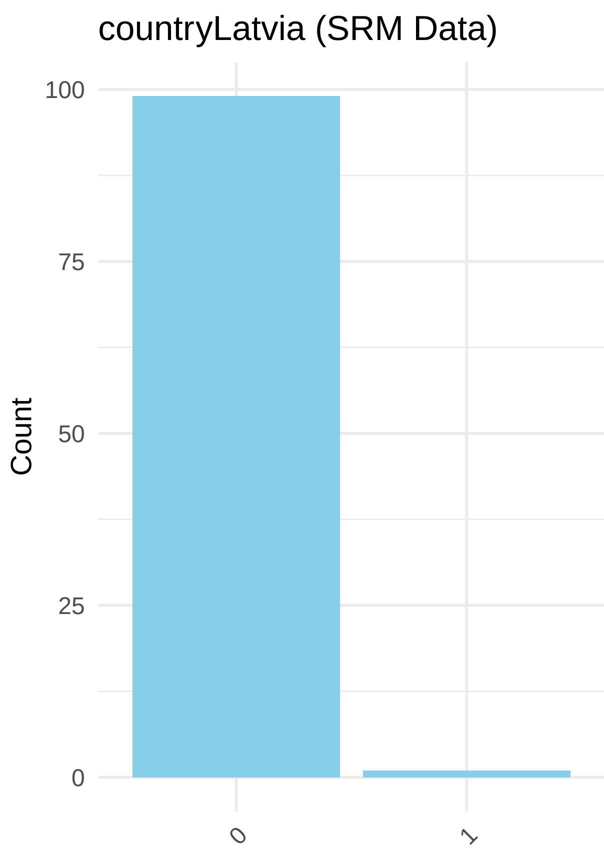

countryHungary (Full Data)
Count
15000
10000
5000
0
0
1



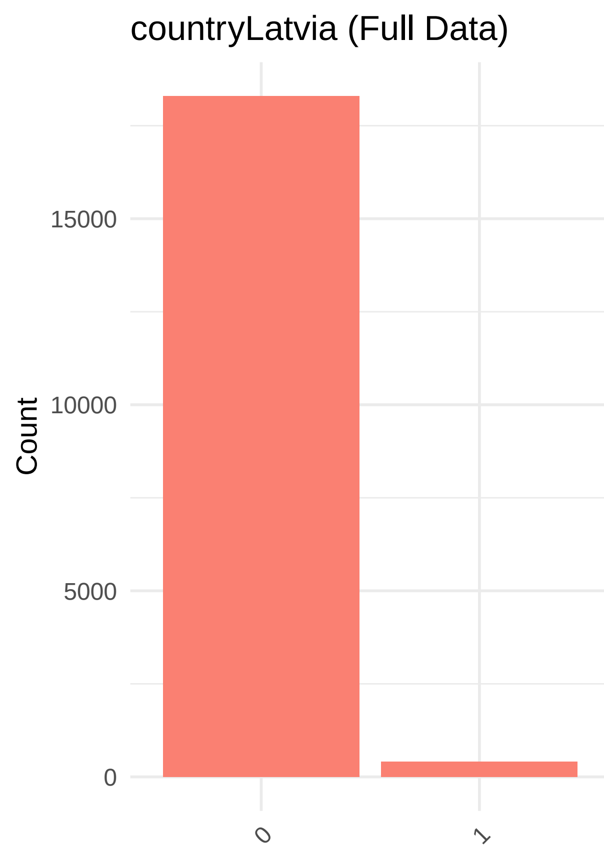
countryIreland (SRM Data)
Count
100
75
50
25
0
0
1
countryIreland (Full Data)
Count
15000
10000
5000
0
0
1



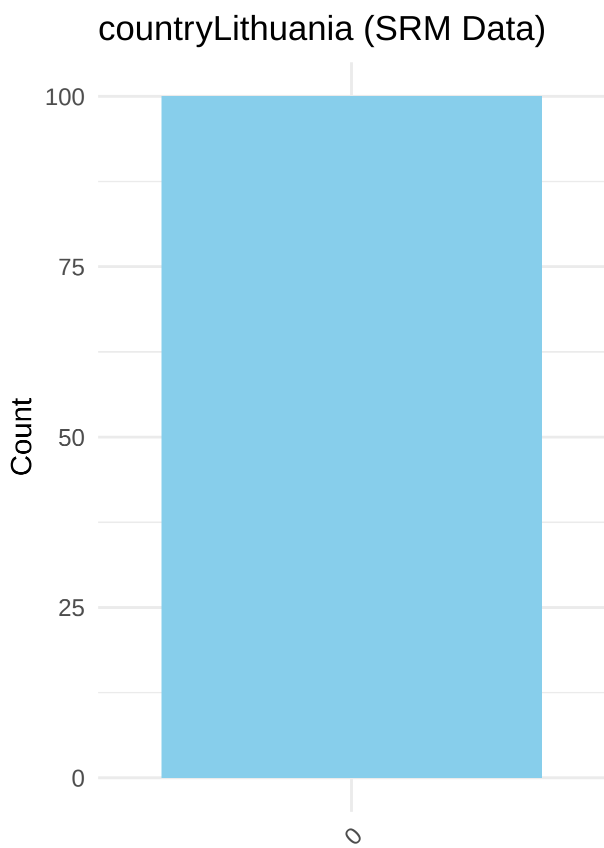
countryItaly (SRM Data)
Count
100
75
50
25
0
0
countryItaly (Full Data)
Count
15000
10000
5000
0
0
1

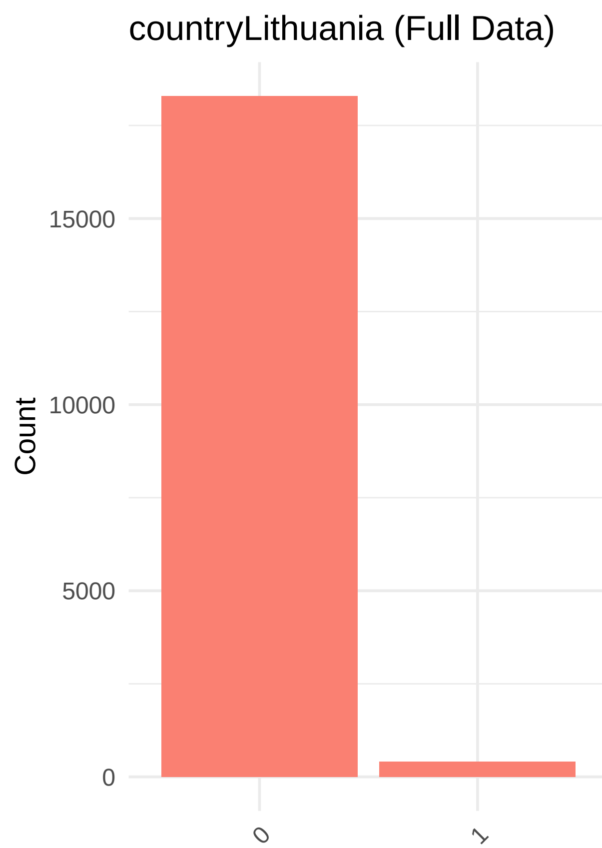


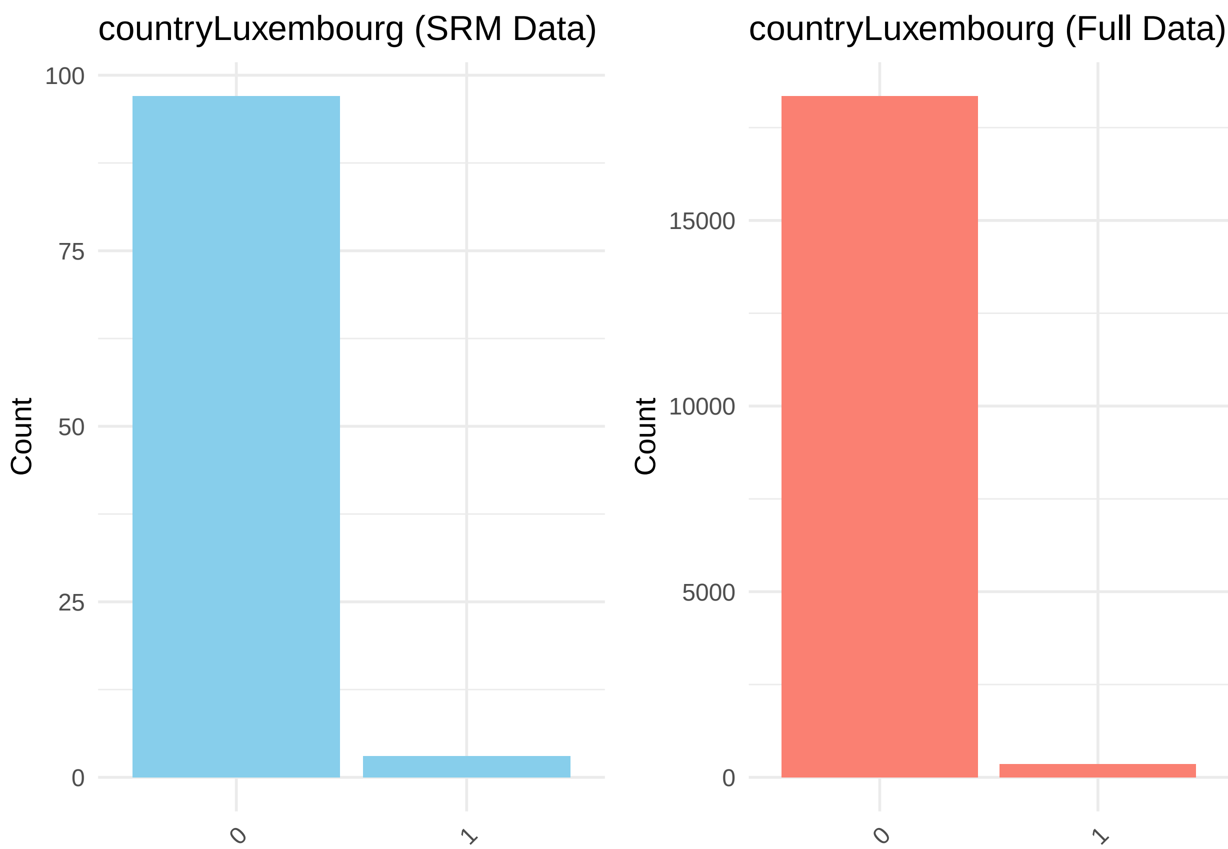
countryLatvia (SRM Data)
100
75
50
25
0
Count
0
1

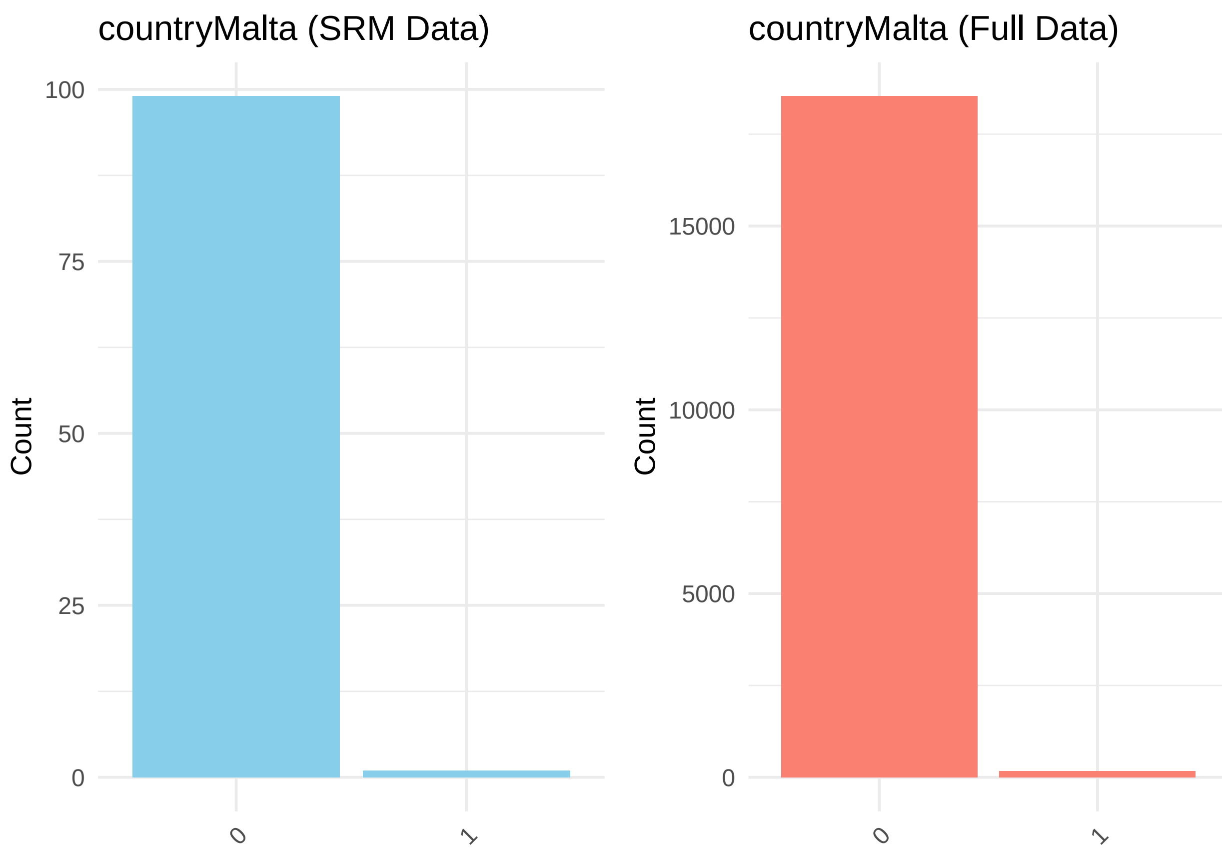


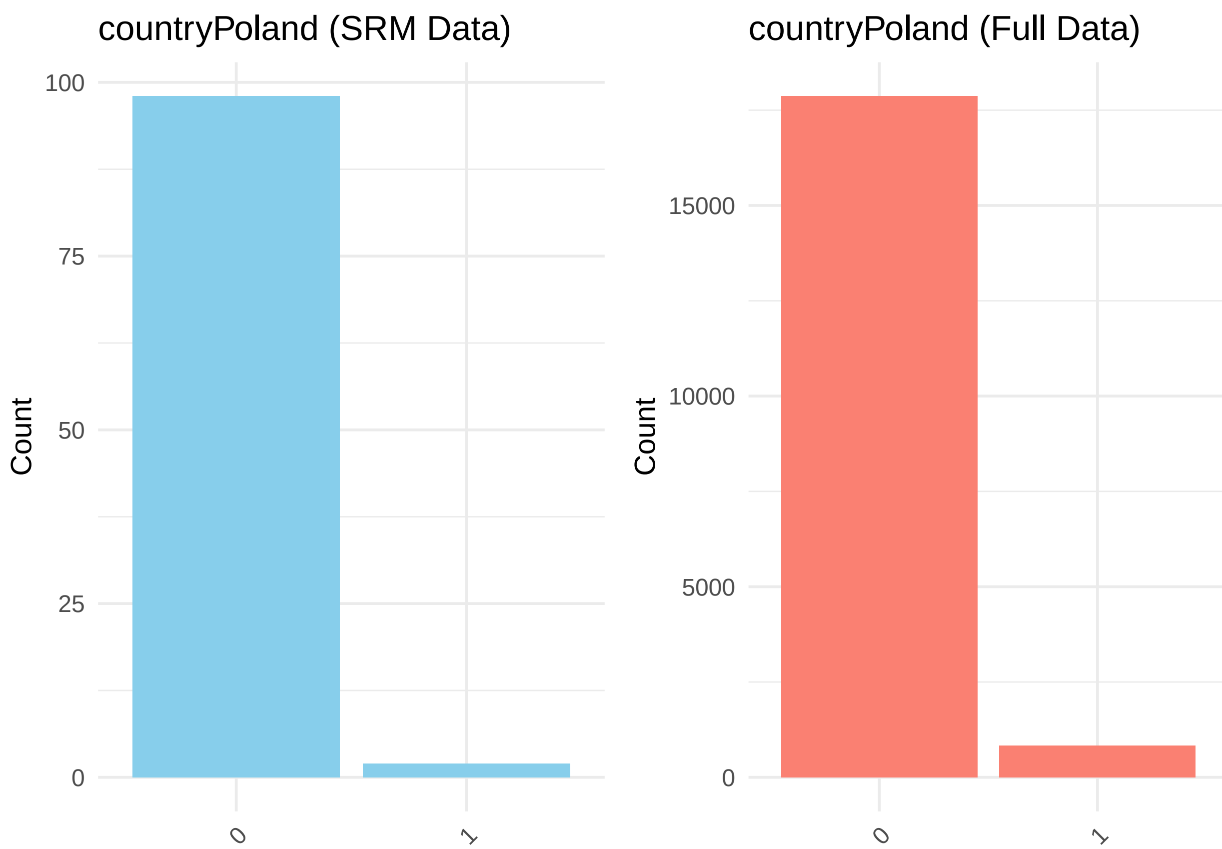
countryLatvia (Full Data)
15000
10000
5000
0
Count
0
1

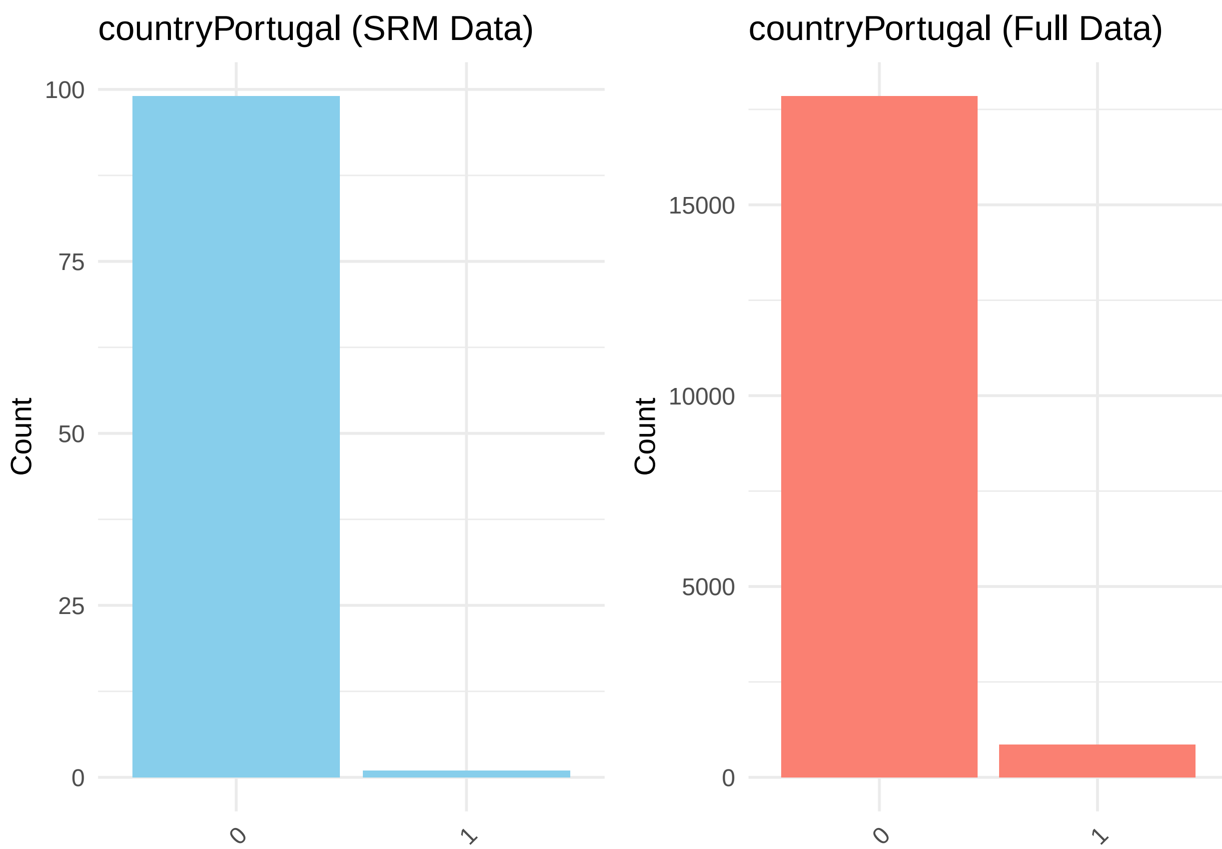

countryLithuania (SRM Data)
Count
100
75
50
25
0
0

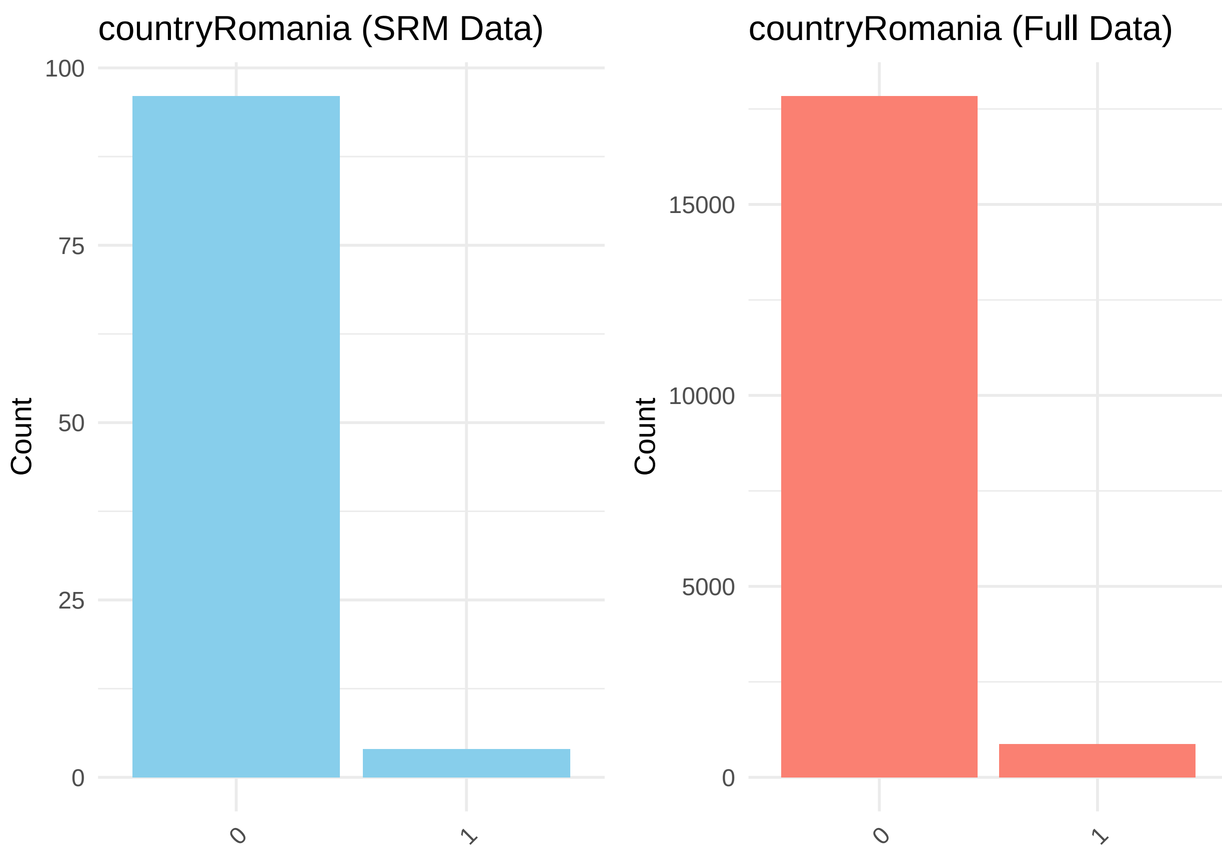

countryLithuania (Full Data)
Count
15000
10000
5000
0
0
1

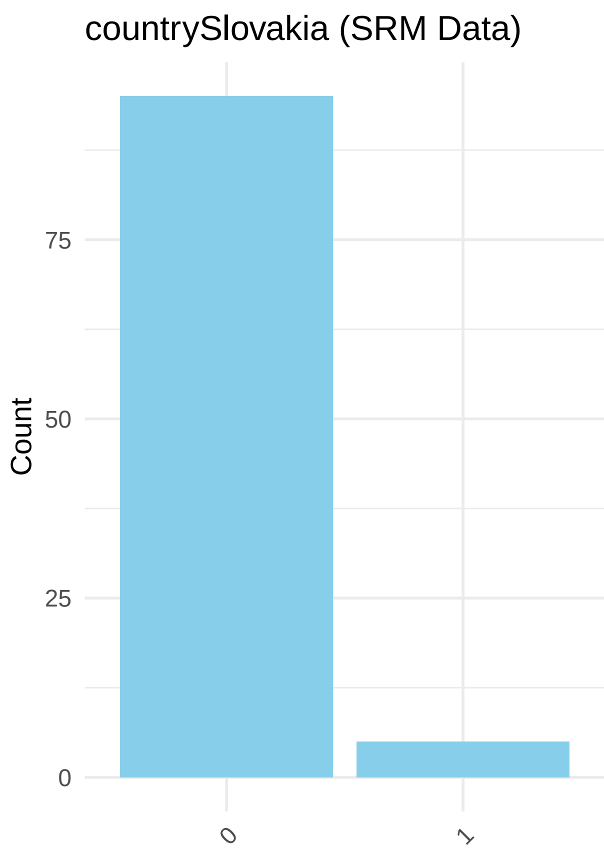

countryLuxembourg (SRM Data)
Count
100
75
50
25
0
0
1
countryLuxembourg (Full Data)
Count
15000
10000
5000
0
0
1

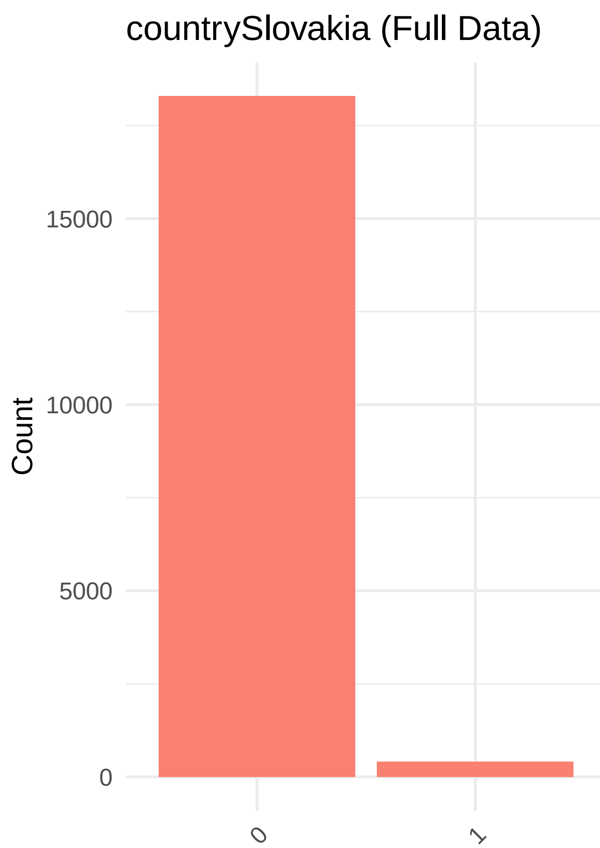

countryMalta (SRM Data)
Count
100
75
50
25
0
0
1
countryMalta (Full Data)
Count
15000
10000
5000
0
0
1


countryPoland (SRM Data)
Count
100
75
50
25
0
0
1
countryPoland (Full Data)
Count
15000
10000
5000
0
0
1

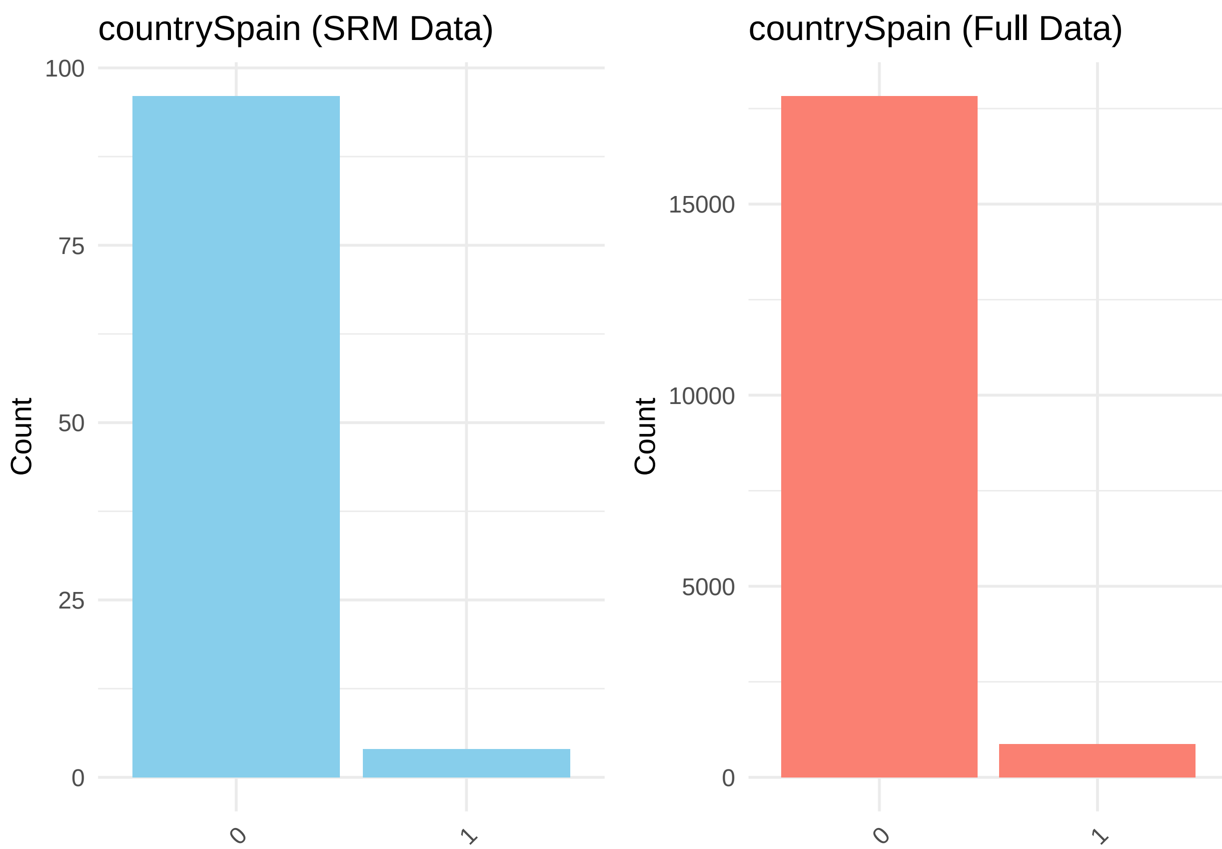

countryPortugal (SRM Data)
Count
100
75
50
25
0
0
1
countryPortugal (Full Data)
Count
15000
10000
5000
0
0
1

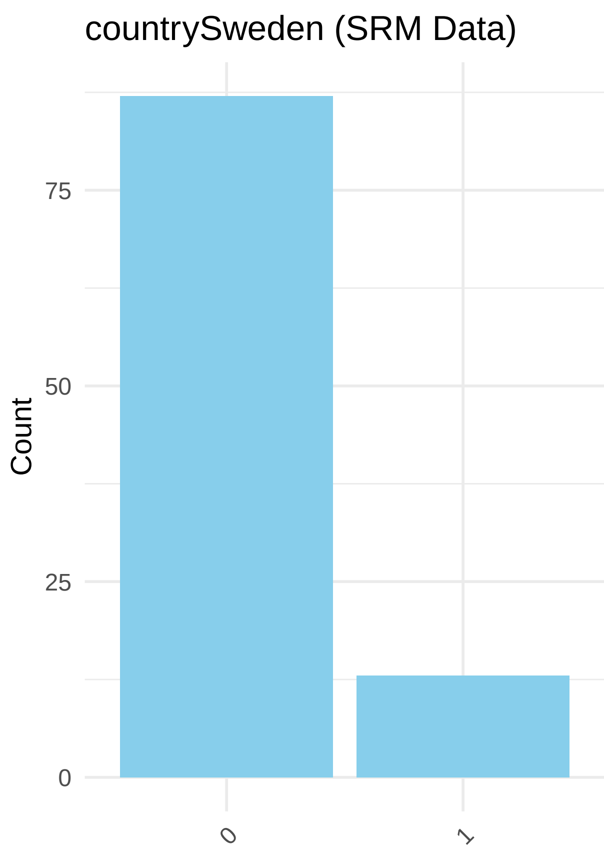

countryRomania (SRM Data)
Count
100
75
50
25
0
0
1
countryRomania (Full Data)
Count
15000
10000
5000
0
0
1

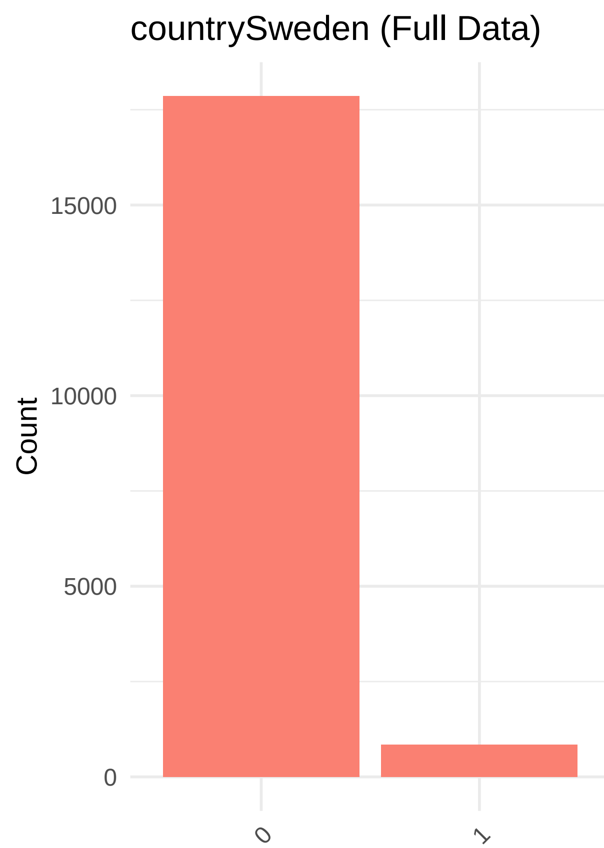

countrySlovakia (SRM Data)
Count
0
25
50
75
0
1

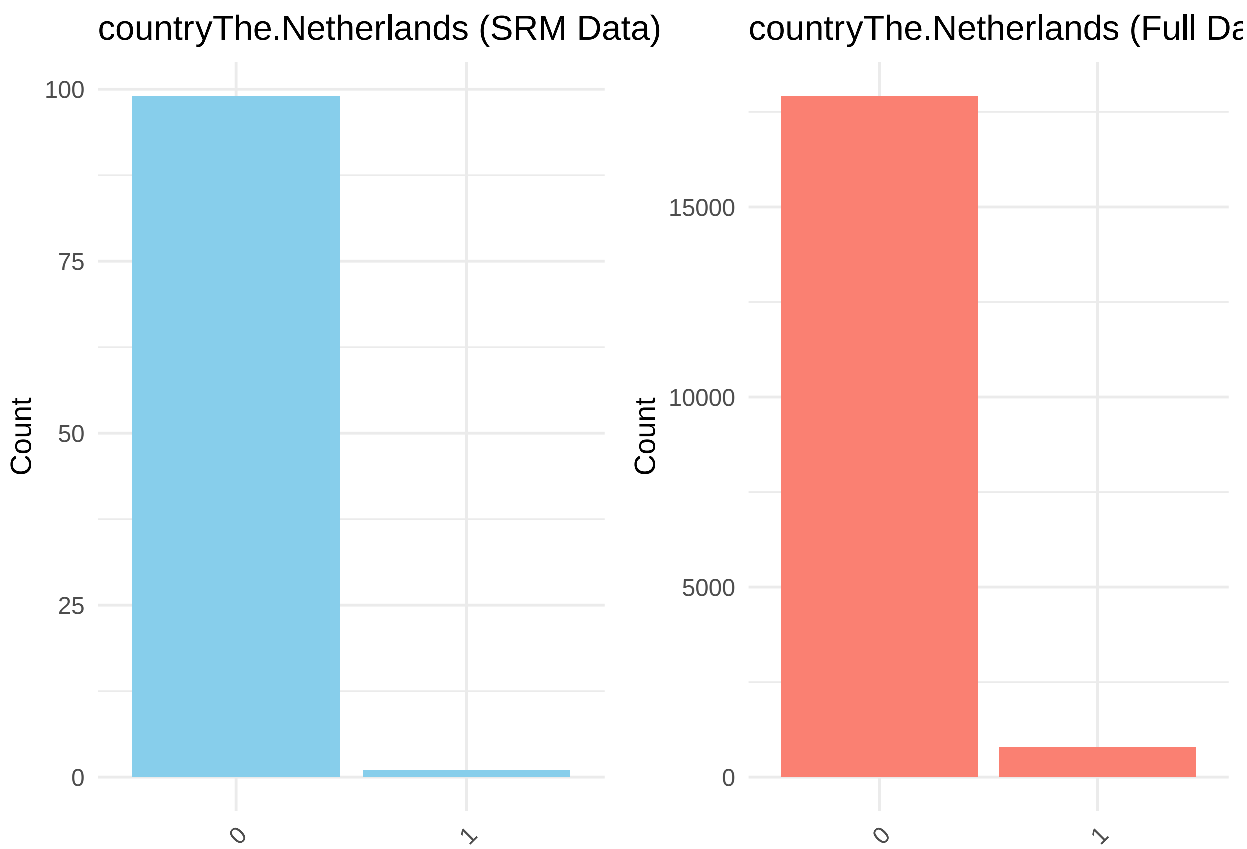

countrySlovakia (Full Data)
Count
0
5000
10000
15000
0
1


countrySlovenia (SRM Data)
countrySlovenia (Full Data)
Count
100
75
50
25
0
15000
10000
5000
1
0


countrySpain (SRM Data)
Count
100
75
50
25
0
0
1
countrySpain (Full Data)
Count
15000
10000
5000
0
0
1


countrySweden (SRM Data)
Count
0
25
50
75
0
1


countrySweden (Full Data)
Count
0
5000
10000
15000
0
1



Creating the SRM critique model:

All possible combiations of the interactions respect to SRM critique is looks like :

. ~ . + age_scale:trust . ~ . + education:trust . ~ . + LR_scale_scale:trust . ~ . + income_scale:trust . ~ . + gender:trust . ~ . + age_scale:trust + education:trust . ~ . + age_scale:trust + LR_scale_scale:trust . ~ . + age_scale:trust + income_scale:trust . ~ . + age_scale:trust + gender:trust . ~ . + education:trust + LR_scale_scale:trust . ~ . + education:trust + income_scale:trust . ~ . + education:trust + gender:trust . ~ . + LR_scale_scale:trust + income_scale:trust . ~ . + LR_scale_scale:trust + gender:trust . ~ . + income_scale:trust + gender:trust . ~ . + age_scale:trust + education:trust + LR_scale_scale:trust . ~ . + age_scale:trust + education:trust + income_scale:trust . ~ . + age_scale:trust + education:trust + gender:trust . ~ . + age_scale:trust + LR_scale_scale:trust + income_scale:trust . ~ . + age_scale:trust + LR_scale_scale:trust + gender:trust . ~ . + age_scale:trust + income_scale:trust + gender:trust . ~ . + education:trust + LR_scale_scale:trust + income_scale:trust . ~ . + education:trust + LR_scale_scale:trust + gender:trust . ~ . + education:trust + income_scale:trust + gender:trust . ~ . + LR_scale_scale:trust + income_scale:trust + gender:trust . ~ . + age_scale:trust + education:trust + LR_scale_scale:trust + income_scale:trust . ~ . + age_scale:trust + education:trust + LR_scale_scale:trust + gender:trust . ~ . + age_scale:trust + education:trust + income_scale:trust + gender:trust . ~ . + age_scale:trust + LR_scale_scale:trust + income_scale:trust + gender:trust . ~ . + education:trust + LR_scale_scale:trust + income_scale:trust + gender:trust . ~ . + age_scale:trust + education:trust + LR_scale_scale:trust + income_scale:trust + gender:trust

```
srm_critique_formula=update(full_formula,. ~ . + age_scale:trust)
```

```r
srm_critique_model=glm(formula =srm_critique_formula ,family = binomial(link = "logit"),data=train_data
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```r
summary(srm_critique_model)
```

```
##
## Call:
## glm(formula = srm_critique_formula, family = binomial(link = "logit"),
##     data = train_data, weights = country_w)
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    0.1324446  0.1092200   1.213  0.22527
## income_scale                  -0.0435607  0.0227317  -1.916  0.05533 .
## trustNo really confident       0.5163152  0.0575343   8.974  < 2e-16 ***
## trustRather confident          1.0413448  0.0639744  16.278  < 2e-16 ***
## trustVery confident            1.3710265  0.0993701  13.797  < 2e-16 ***
## LR_scale_scale                -0.0792614  0.0099683  -7.951 1.84e-15 ***
## new_ccknowledge_index_scale    3.3359968  0.1392912  23.950  < 2e-16 ***
## residencetown                 -0.0519311  0.0483008  -1.075  0.28230
## residencerural                -0.1798561  0.0575040  -3.128  0.00176 **
## age_scale                     -0.0004908  0.0030227  -0.162  0.87102
## gendermale                    -0.0363312  0.0428705  -0.847  0.39674
## educationprimary               0.0972120  0.0629674   1.544  0.12263
## educationtertiary              0.0917967  0.0471744   1.946  0.05167 .
## has_childrenyes                0.0620776  0.0469028   1.324  0.18566
## countryBelgium                 0.3550353  0.1312815   2.704  0.00684 **
## countryBulgaria                0.9359661  0.1409107   6.642 3.09e-11 ***
## countryCroatia                 1.0805284  0.1410249   7.662 1.83e-14 ***
## countryCyprus                  1.1918366  0.2164242   5.507 3.65e-08 ***
## countryCzech.Republic          0.3511166  0.1279172   2.745  0.00605 **
## countryDenmark                 0.5881633  0.1360676   4.323 1.54e-05 ***
## countryEstonia                -0.0295715  0.1502871  -0.197  0.84401
## countryFinland                 0.3432525  0.1300498   2.639  0.00831 **
## countryFrance                  0.5448647  0.1362616   3.999 6.37e-05 ***
## countryGermany                -0.0056905  0.1217131  -0.047  0.96271
## countryGreece                  1.3256636  0.1529074   8.670  < 2e-16 ***
## countryHungary                 0.3169152  0.1301902   2.434  0.01492 *
## countryIreland                 0.9571435  0.1399371   6.840 7.93e-12 ***
## countryItaly                   0.8020915  0.1393301   5.757 8.57e-09 ***
## countryLatvia                  0.2045608  0.1553457   1.317  0.18790
## countryLithuania               0.1558939  0.1570157   0.993  0.32078
## countryLuxembourg              0.4704085  0.1753281   2.683  0.00730 **
## countryMalta                   2.2427501  0.4248783   5.279 1.30e-07 ***
## countryPoland                  0.2068465  0.1281218   1.614  0.10643
## countryPortugal                1.3528302  0.1517026   8.918  < 2e-16 ***
## countryRomania                 0.5629941  0.1289535   4.366 1.27e-05 ***
## countrySlovakia                0.2108706  0.1545533   1.364  0.17245
## countrySlovenia                0.9014158  0.1747458   5.158 2.49e-07 ***
## countrySpain                   0.6076286  0.1326990   4.579 4.67e-06 ***
## countrySweden                  0.0796602  0.1253699   0.635  0.52517
```

```
## countryThe.Netherlands              0.3248200  0.1305375   2.488  0.01283 *
## trustNo really confident:age_scale  0.0080621  0.0035162   2.293  0.02186 *
## trustRather confident:age_scale     0.0049407  0.0038273   1.291  0.19673
## trustVery confident:age_scale      -0.0078091  0.0058785  -1.328  0.18404
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15803  on 14958  degrees of freedom
## Residual deviance: 14302  on 14916  degrees of freedom
## AIC: 14916
##
## Number of Fisher Scoring iterations: 5
```

```
yhat_updated_train=predict(srm_critique_model,newdata=train_data,type="response")
yhat_updated_test=predict(srm_critique_model,newdata=test_data,type="response")

class_pred_updated_train <- ifelse(yhat_updated_train > 0.5, 1, 0)
class_pred_updated_test  <- ifelse(yhat_updated_test > 0.5, 1, 0)
```

Checking the Accuracy Metrics respect to the Train Data for the Updated Model:

```
confusionMatrix(as.factor(train_data$ctax_binary),as.factor(class_pred_updated_train))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0   438  2931
##          1   260 11330
##
##                Accuracy : 0.7867
##                  95% CI : (0.78, 0.7932)
##     No Information Rate : 0.9533
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1497
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.62751
##             Specificity : 0.79447
##          Pos Pred Value : 0.13001
##          Neg Pred Value : 0.97757
##              Prevalence : 0.04666
##          Detection Rate : 0.02928
##    Detection Prevalence : 0.22522
##       Balanced Accuracy : 0.71099
```

```
##
##        'Positive' Class : 0
##
```

```r
confusionMatrix(as.factor(test_data$ctax_binary),as.factor(class_pred_updated_test))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  117  733
##          1   51 2838
##
##                Accuracy : 0.7903
##                  95% CI : (0.7769, 0.8033)
##     No Information Rate : 0.9551
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1674
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.69643
##             Specificity : 0.79474
##          Pos Pred Value : 0.13765
##          Neg Pred Value : 0.98235
##              Prevalence : 0.04493
##          Detection Rate : 0.03129
##    Detection Prevalence : 0.22733
##       Balanced Accuracy : 0.74558
##
##        'Positive' Class : 0
##
```

```r
accuracy_df <- data.frame(
  model = c("Baseline", "SRM Critique", "ML Model"),
  accuracy = c(
    mean(class_pred_test == test_data$ctax_binary, na.rm = TRUE),
    mean(class_pred_updated_test == test_data$ctax_binary, na.rm = TRUE),
    mean(yhat_rf_test == test_data$ctax_binary, na.rm = TRUE)
  )
)
```

```r
accuracy_df$label <- substr(as.character(accuracy_df$accuracy), 1, 6)


library(ggplot2)

ggplot(accuracy_df, aes(x = model, y = accuracy, fill = model)) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = label), vjust = -0.5) +
  labs(
    title = "Model Accuracy Comparison",
```

```r
    x = "Model",
    y = "Accuracy"
  ) +
  scale_fill_manual(values = c(
    "Baseline" = "lightgreen",
    "Baseline + SRM" = "steelblue",
    "ML Model" = "darkgreen"
  )) +
  ylim(0, 1) +
  theme_minimal()
```