EPH_Dataset

Horacio Báez

20/6/2020

Preface

This report is part of the project Data Science: Capstone of the HarvardX's Data Science Profesional Certificate Program.

Introduction

Paraguay is a small country in the center of South America that has been growing a lot since the beginning of the 2000's. However, after almost 20 years of sustained growth, the Government still has a debt with public education and school dropout.

Even nowadays, just around 10% of the population have the opportunity to get into college and even less people finish an undergraduate degree. This is caused mostly by two factors: cost of studying in the University and highschool dropout. In this project, we will focus mainly on highschool dropout.

EPH Dataset

The General Directorate of Statistics (*DGEEC*, by its spanish acronym) set up the House's Permanent Poll or *Encuesta Permanente de Hogares* (*EPH*, by its spanish acronym) collects information about a wide range of social and economic issues. One of the dimension of this Poll, is related to education.

The goal of this project is to predict highschool dropout based on data from the *EPH*.

We will be using the *EPH* of the year 2017 for training purposes and teh *EPH* of the year 2018 to test the model.

Model Evaluation

The two metric that we will use to evaluate the performance of the model are the *sensitivity* (also known as *recall*) and the *F1 score*.

$Sensitivity = \frac{TP}{\mathrm{TP + FN}}$

Where TP stands for *True Positive* and FN stands for *False Negatives*.

$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$

Where $recall = sensitivity$ and $precision = \frac{TP}{TP+FP}$ with FP standing for *False Positives*.

Process and workflow

The main steps in any data science project consists of:

1.Data preparation: dowloading the dataset and prepare it to be processed and analysed.

2.Data exploration and visualization: explore the data to understand the relantionship between variables.

3.Data Cleaning: the dataset may contain data that is not relevant that can be removed or may need some other tpy of transformation before modeling.

4.Data analysis and modeling: create a model using the insights obtained in the step 2. Then, test the model.

5.Communicate: reporting the results.

First, we load the 2017's *EPH* as `eph_train` and the 2018's *EPH* as `eph_test`.

After that first step, we proceed to explore the dataset. We can use the dictionary to ease our exploration, but the `View()` adn `head()` functions are useful as well. In this step we are going to plot some variables and explore them to find insights that will help us later to build our model of dropout prediction.

For training our model, we are going to use the `train()` function of the `caret` package, that is already optimize to try out different models with different models. At last we going to create a `results` data frame with the `sensitivity` and `F1_score` of the different models.

Method and analysis

Data Preparation

In this section we will load both datasets for training (`eph_train`) and testing (`eph_test`) and load the packages needed throughout the project.

```
if(!require(tidyverse)) install.packages("tidyverse", repos="http://cran.us.r-project.org")
if(!require(haven)) install.packages("haven", repos="http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos="http://cran.us.r-project.org")
if(!require(MLmetrics)) install.packages("MLmetrics", repos="http://cran.us.r-project.org")
if(!require(ggthemes)) install.packages("ggthemes", repos="http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos="http://cran.us.r-project.org")
#============================================================
eph_train <- read_sav("REG02_EPHC_ANUAL_2017.SAV")

eph_test <- read_sav("REG02_EPHC_ANUAL_2018.SAV")
```

Data Exploration

Now that we have loaded the datasets and packaged required, we can start to explore the dataset.

```
class(eph_train)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
class(eph_test)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
head(eph_train)
```

```
## # A tibble: 6 x 218
##     UPM NVIVI NHOGA TRIMESTRE        AÑO    DPTO    AREA   LO2   P02      P03
##   <dbl> <dbl> <dbl> <dbl+lbl>  <dbl+lbl> <dbl+l> <dbl+l> <dbl> <dbl> <dbl+lb>
## 1    21     8     1 2 [Segun~ 2017 [Año~ 0 [Asu~ 1 [Urb~     1    39  1 [Jef~
## 2    21     8     1 2 [Segun~ 2017 [Año~ 0 [Asu~ 1 [Urb~     2    23  3 [Hij~
## 3    21     8     1 2 [Segun~ 2017 [Año~ 0 [Asu~ 1 [Urb~     3    17  3 [Hij~
## 4    21    14     1 2 [Segun~ 2017 [Año~ 0 [Asu~ 1 [Urb~     1    78  1 [Jef~
## 5    21    14     1 2 [Segun~ 2017 [Año~ 0 [Asu~ 1 [Urb~     2    41  3 [Hij~
## 6    21    14     1 2 [Segun~ 2017 [Año~ 0 [Asu~ 1 [Urb~     3    68 11 [Per~
## # ... with 208 more variables: P04 <dbl+lbl>, P04A <dbl+lbl>, P04B <dbl+lbl>,
## #   P05C <dbl+lbl>, P05P <dbl+lbl>, P05M <dbl+lbl>, P06 <dbl+lbl>,
## #   P08D <dbl+lbl>, P08M <dbl+lbl>, P08A <dbl+lbl>, P09 <dbl+lbl>,
## #   A01 <dbl+lbl>, A01A <dbl+lbl>, A02 <dbl+lbl>, A03 <dbl+lbl>, A04 <dbl+lbl>,
## #   A04A <dbl+lbl>, A05 <dbl+lbl>, A07 <dbl+lbl>, A08 <dbl+lbl>, A10 <dbl+lbl>,
## #   A11A <dbl+lbl>, A11M <dbl+lbl>, A11S <dbl+lbl>, A12 <dbl+lbl>,
## #   A13REC <dbl+lbl>, A14REC <dbl+lbl>, A15 <dbl+lbl>, A16 <dbl+lbl>,
## #   A17A <dbl+lbl>, A17M <dbl+lbl>, A17S <dbl+lbl>, A18 <dbl+lbl>,
## #   B01REC <dbl+lbl>, B02REC <dbl+lbl>, B03LU <dbl+lbl>, B03MA <dbl+lbl>,
## #   B03MI <dbl+lbl>, B03JU <dbl+lbl>, B03VI <dbl+lbl>, B03SA <dbl+lbl>,
## #   B03DO <dbl+lbl>, B04 <dbl+lbl>, B05 <dbl+lbl>, B06 <dbl+lbl>,
## #   B07A <dbl+lbl>, B07M <dbl+lbl>, B07S <dbl+lbl>, B08 <dbl+lbl>,
## #   B09A <dbl+lbl>, B09M <dbl+lbl>, B09S <dbl+lbl>, B10 <dbl+lbl>,
## #   B11 <dbl+lbl>, B12 <dbl+lbl>, B12A <dbl+lbl>, B12B <dbl+lbl>,
## #   B12C <dbl+lbl>, B13 <dbl+lbl>, B14 <dbl+lbl>, B15 <dbl+lbl>,
## #   B16G <dbl+lbl>, B16U <dbl+lbl>, B16D <dbl+lbl>, B16T <dbl+lbl>,
## #   B17 <dbl+lbl>, B18AG <dbl+lbl>, B18AU <dbl+lbl>, B18BG <dbl+lbl>,
## #   B18BU <dbl+lbl>, B19 <dbl+lbl>, B20G <dbl+lbl>, B20U <dbl+lbl>,
## #   B20D <dbl+lbl>, B20T <dbl+lbl>, B21 <dbl+lbl>, B22 <dbl+lbl>,
## #   B23 <dbl+lbl>, B24 <dbl+lbl>, B25 <dbl+lbl>, B26 <dbl+lbl>, B271 <dbl+lbl>,
## #   B272 <dbl+lbl>, B28 <dbl+lbl>, B29 <dbl+lbl>, B30 <dbl+lbl>, B31 <dbl+lbl>,
## #   C01REC <dbl+lbl>, C02REC <dbl+lbl>, C03 <dbl+lbl>, C04 <dbl+lbl>,
## #   C05 <dbl+lbl>, C06 <dbl+lbl>, C07 <dbl+lbl>, C08 <dbl+lbl>, C09 <dbl+lbl>,
## #   C101 <dbl+lbl>, C102 <dbl+lbl>, C11G <dbl+lbl>, C11U <dbl+lbl>, ...
```

```
dim(eph_train)
```

```
## [1] 73643   218
```

We can see that, although we have less that 100,000 observations, the dataset contains 218 variables, so we will not cover or use all of them in this project since many of them have no predictive power. Despite we will not use them, they may still be useful in other projects in which the objective it is different.

If we dig in enough, we can realize that there is no variable that indicates wheter a person has dropped out or not. We are going to add a `desertor` variable later.

Income-related variables

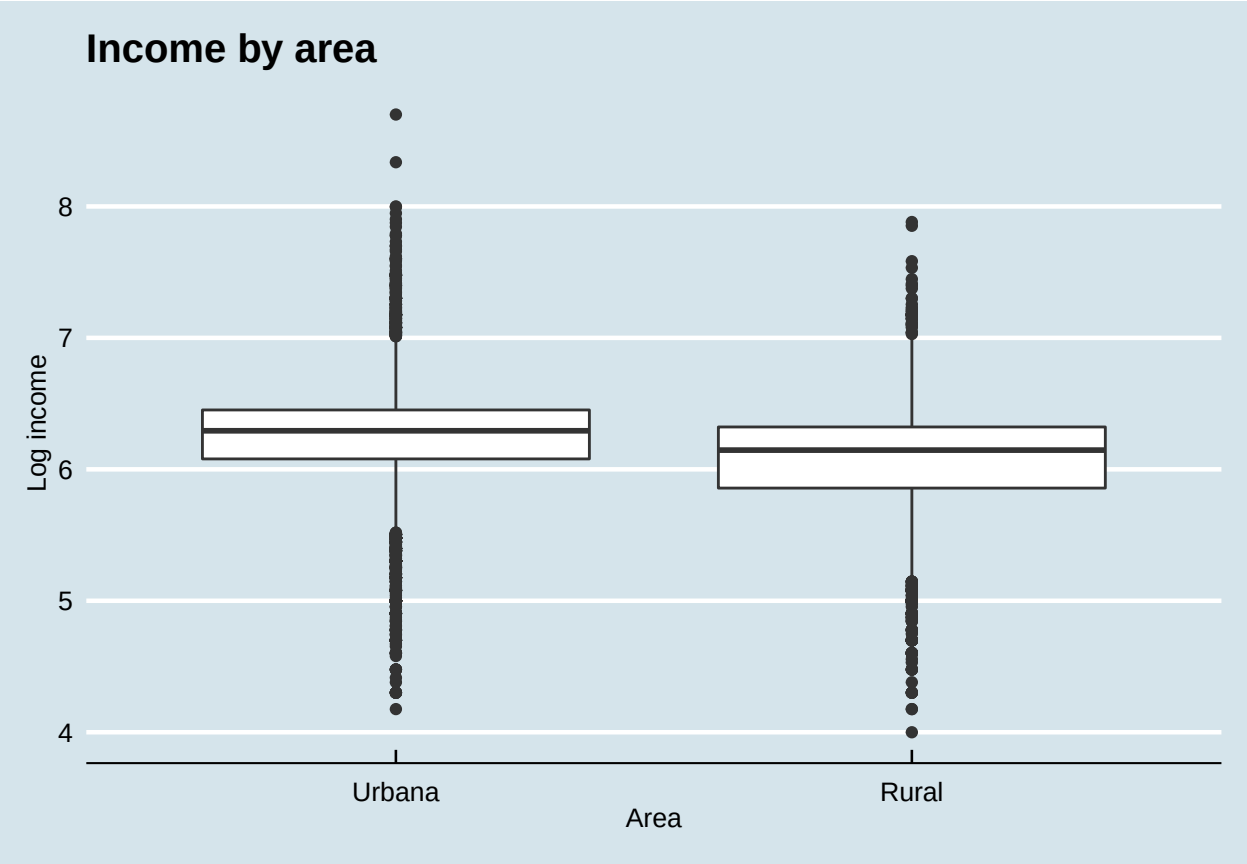
First, let us plot some variables related to the people's income and its distribution.

```r
#Income distribution by department
eph_train%>%
  filter(as.numeric(E01A)<999999999)%>%
  filter(E01A!=0)%>%
  group_by(DPTO)%>%
  ggplot(aes(x=as_factor(DPTO), y=log10(as.numeric(E01A))))+
  geom_boxplot()+
  coord_flip()+
  ylab("Log income")+
  xlab("Department")+
  ggtitle("Income by department")+
  theme_economist()
```

It is important to note that Paraguay's territory splits into 17 departments and an autonomous capital city (Asuncion). The territory of these departments can be tagged into two categories: urban and rural. Let us explore the income by area.

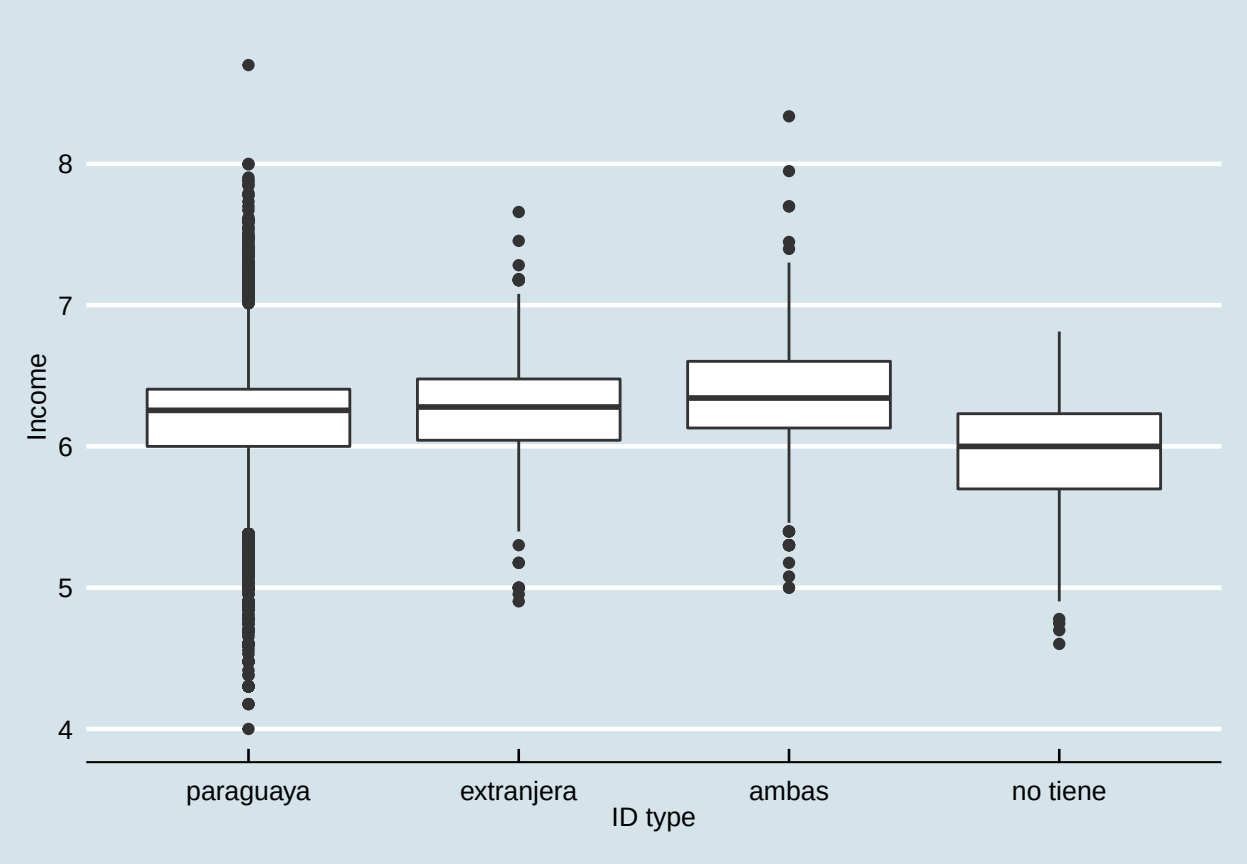
```r
#Income distribution by area
eph_train%>%
  filter(as.numeric(E01A)<999999999)%>%
  filter(E01A!=0)%>%
  group_by(DPTO)%>%
  ggplot(aes(x=as_factor(AREA), y=log10(as.numeric(E01A))))+
  geom_boxplot()+
  ylab("Log income")+
  xlab("Area")+
```

```
  ggtitle("Income by area")+
  theme_economist()
```



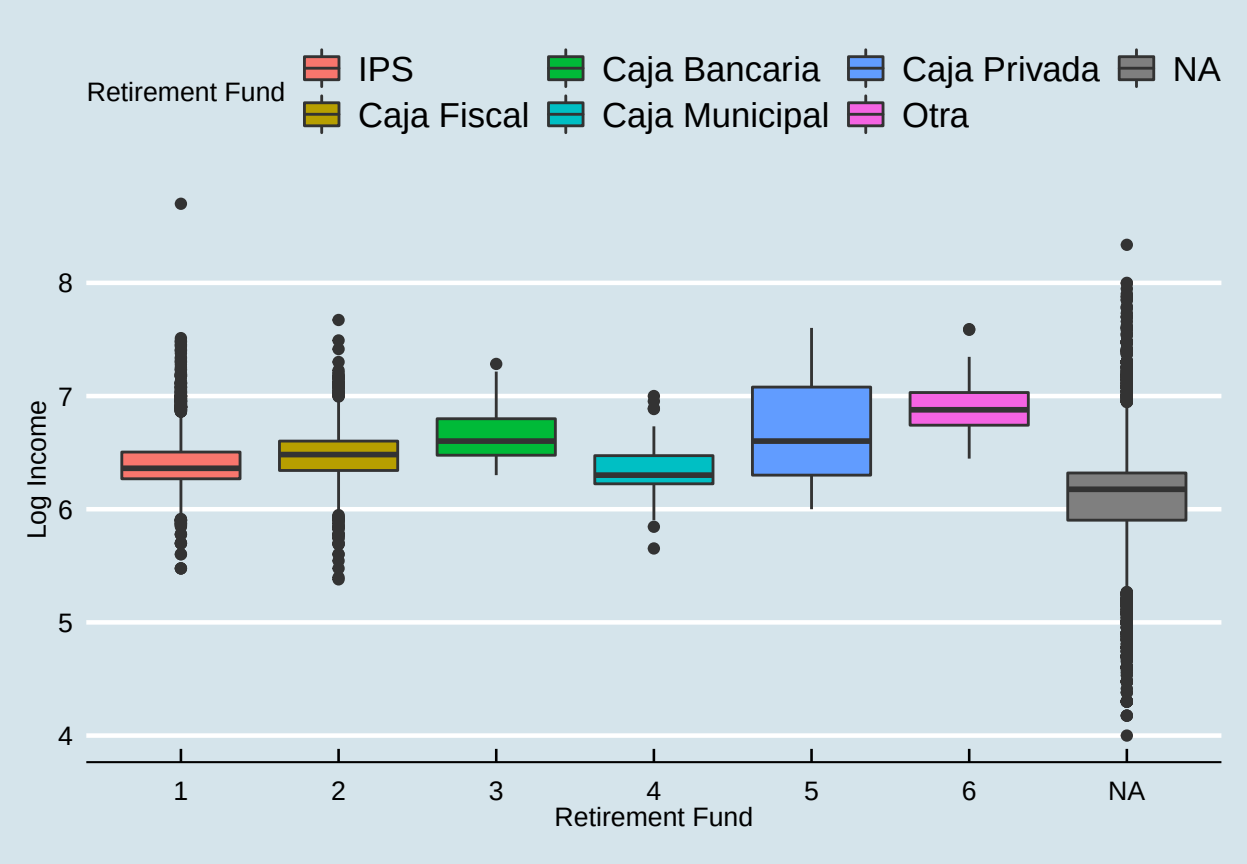
Another interesting fact is that, even nowadays, not everyone has an ID card and that may influence the job type of the people and, therefore, their income.

```
library(tidyverse)
library(ggthemes)
#Income by ID type
eph_train%>%
  filter(E01A<999999999)%>%
  filter(E01A!=0)%>%
  group_by(P04A)%>%
  ggplot(aes(y=log10(as.numeric(E01A)), x=(as_factor(P04A))))+
  geom_boxplot()+
  ylab("Income")+
  xlab("ID type")+
  theme_economist()
```



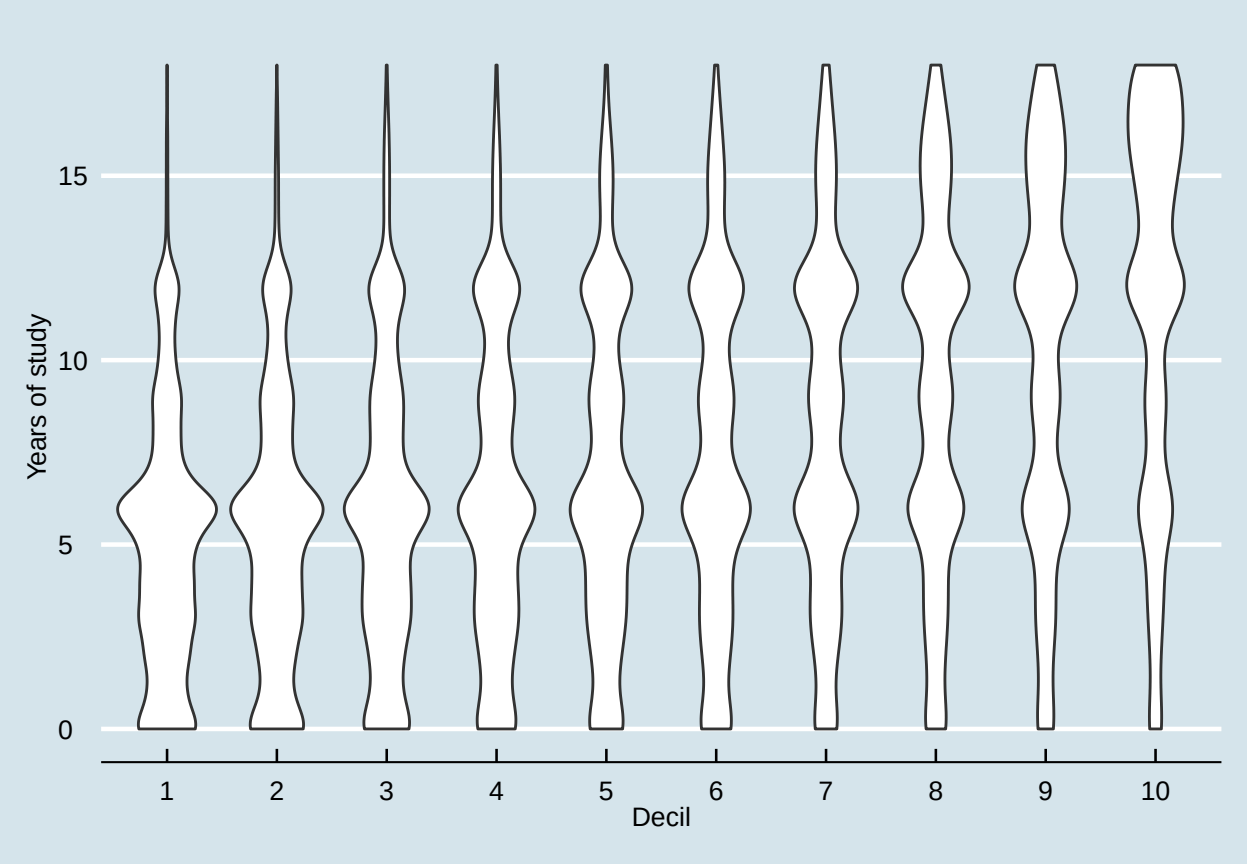
Another important issue that we can mention but will not be the focus of this report is the retirement of the people. The Government could not yet offer a complete solution so many people just do not have any retirement fund.

```
#Income by retirement fund
eph_train%>%
  filter(E01A<999999999)%>%
  filter(E01A!=0)%>%
  group_by(B11)%>%
  ggplot(aes(y=log10(as.numeric(E01A)), x=as.factor(B11)))+
  geom_boxplot(aes(fill=as_factor(B11)))+
  ylab("Log Income")+
  xlab("Retirement Fund")+
  labs(fill="Retirement Fund")+
  theme_economist()
```



A common idea is that the income is directly linked to the amount of years of study. Here we plot that.

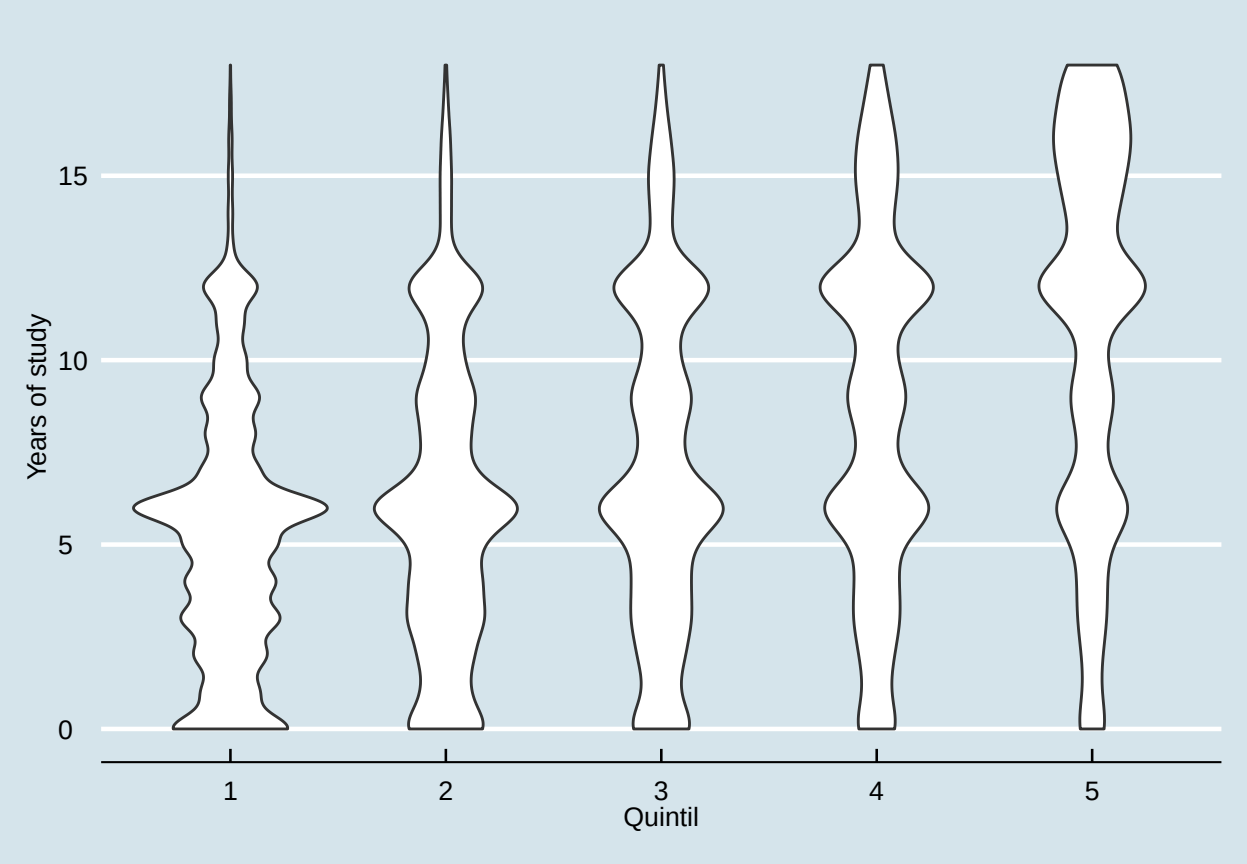
```
#Years of study by decil of income
eph_train%>%
  filter(añoest%in%c('0',"1","2","3","4",'5','6','7','8','9','10','11','12','13','14','15','16','17','18
  filter(decili%in%c('0',"1","2","3","4",'5','6','7','8','9','10'))%>%
  ggplot(aes(y=as.numeric(añoest),x=as.factor(decili)))+
  geom_violin()+
  ylab("Years of study")+
  xlab("Decil")+
  theme_economist()
```



We can notice that the lower the income, less chances of finishing highschool. The meaning of "finishing highschool" will be explained later, but a first graphic approach is useful.

We can plot the same as before, but now with quintiles.

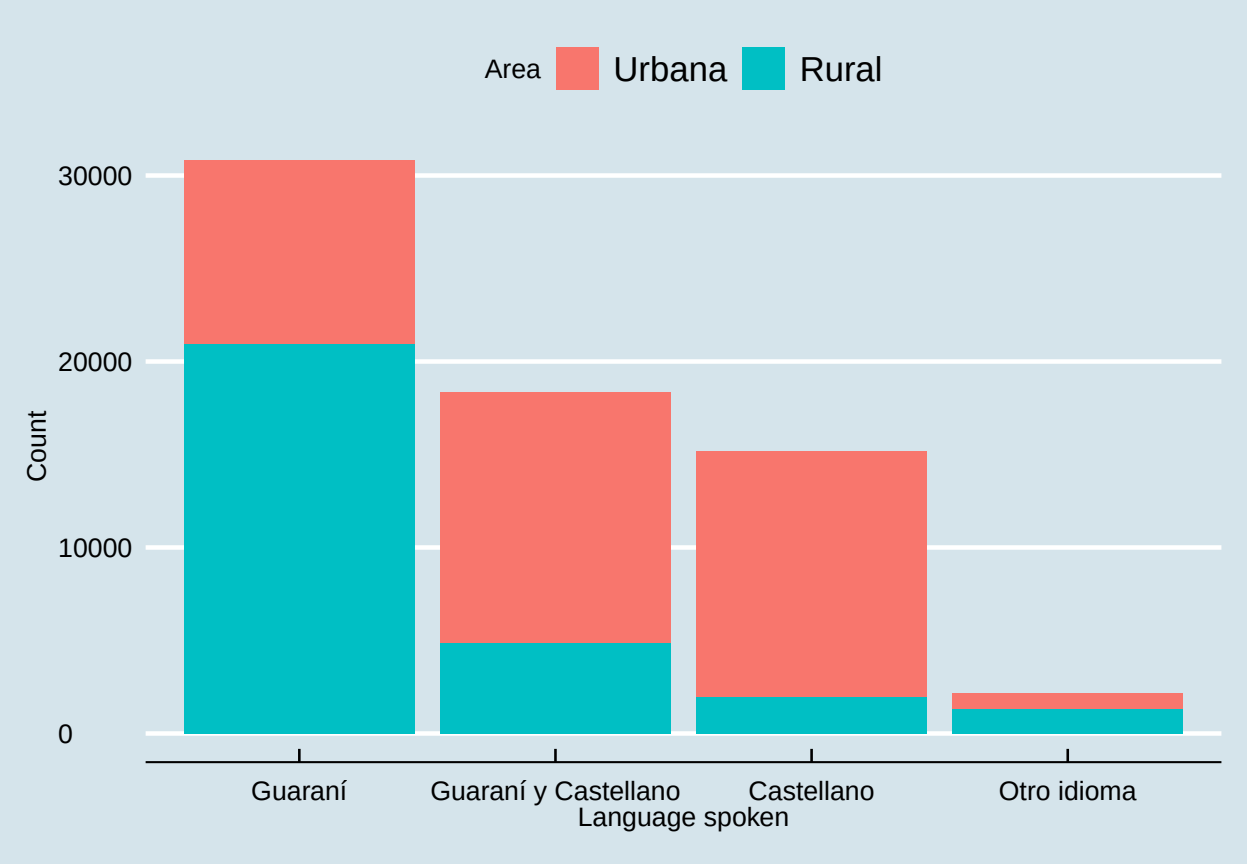
```
#Years of study by quintile
eph_train%>%
  filter(añoest%in%c('0',"1","2","3","4",'5','6','7','8','9','10','11','12','13','14','15','16','17','18
  filter(quintili%in%c('0',"1","2","3","4",'5'))%>%
  ggplot(aes(y=as.numeric(añoest),x=as.factor(quintili)))+
  geom_violin()+
  ylab("Years of study")+
  xlab("Quintil")+
  theme_economist()
```



Language-related variables

Now, before proceeding, it is essential to point out that Paraguay has two oficial languages: spanish and guarani. However, not everyone speaks both languages and this has several consecuences in the educational system. Let us explore that.

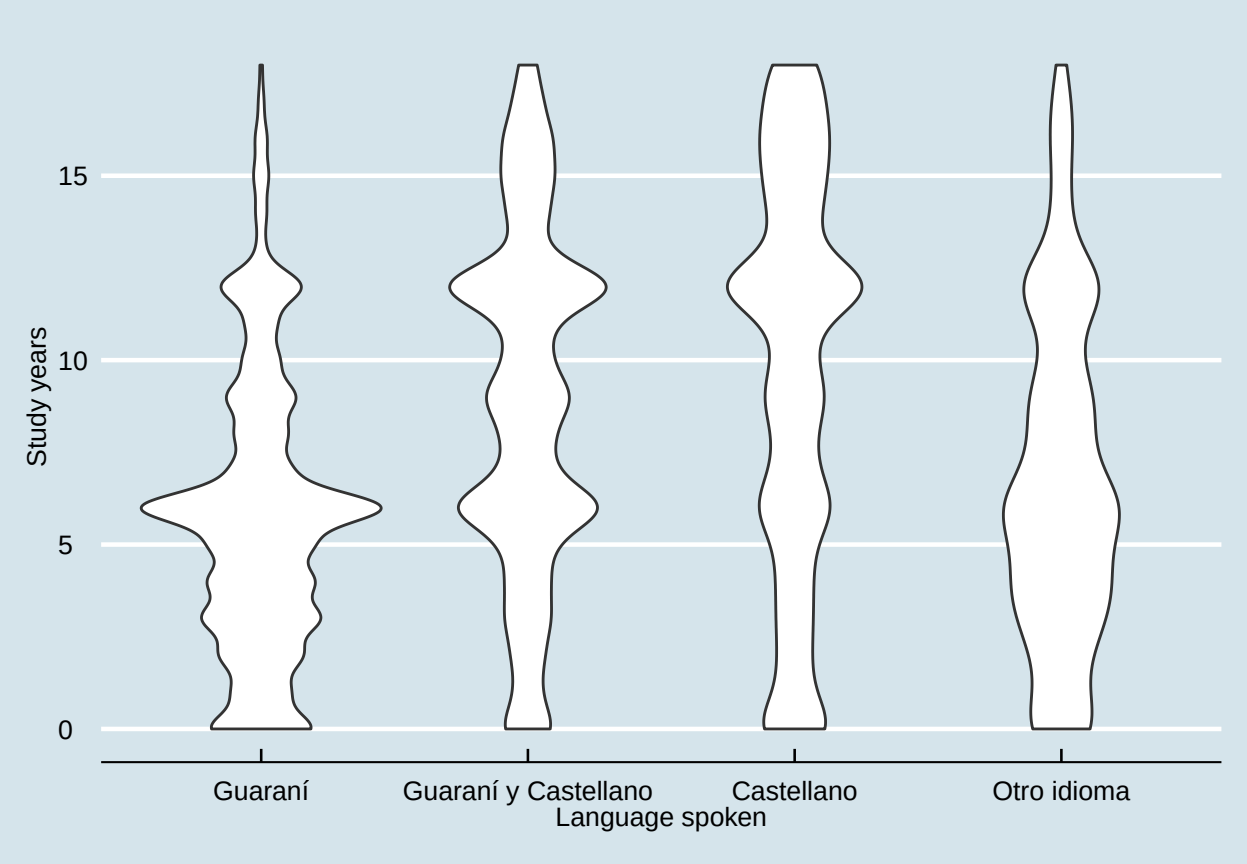
```
#Language spoken by area
eph_train%>%
  filter(ED01%in%c("1","2","3","4"))%>%
  ggplot(aes(x=as_factor(ED01)))+
  geom_bar(aes(fill=as_factor(AREA)))+
  ylab("Count")+
  xlab("Language spoken")+
  labs(fill="Area")+
  theme_economist()
```



We can notice that guarani is the common language in rural areas, while spanish has a higher prevalence in urban areas.

As shown in the graphic below, the fact that guarani is your main language, make you less likely to finish highschool. This is due to the fact that almost everything in the education system was developed in spanish and for spanish learners.

```
#Years of study by language spoken
eph_train%>%
  filter(ED01%in%c("1","2","3","4"))%>%
  filter(añoest%in%c('0','1","2","3","4",'5','6','7','8','9','10',
                     '11','12','13','14','15','16','17','18'))%>%
  group_by(ED01)%>%
  ggplot(aes(x=as_factor(ED01), y=as.numeric(añoest)))+
  geom_violin()+
  ylab("Study years")+
  xlab("Language spoken")+
  theme_economist()
```



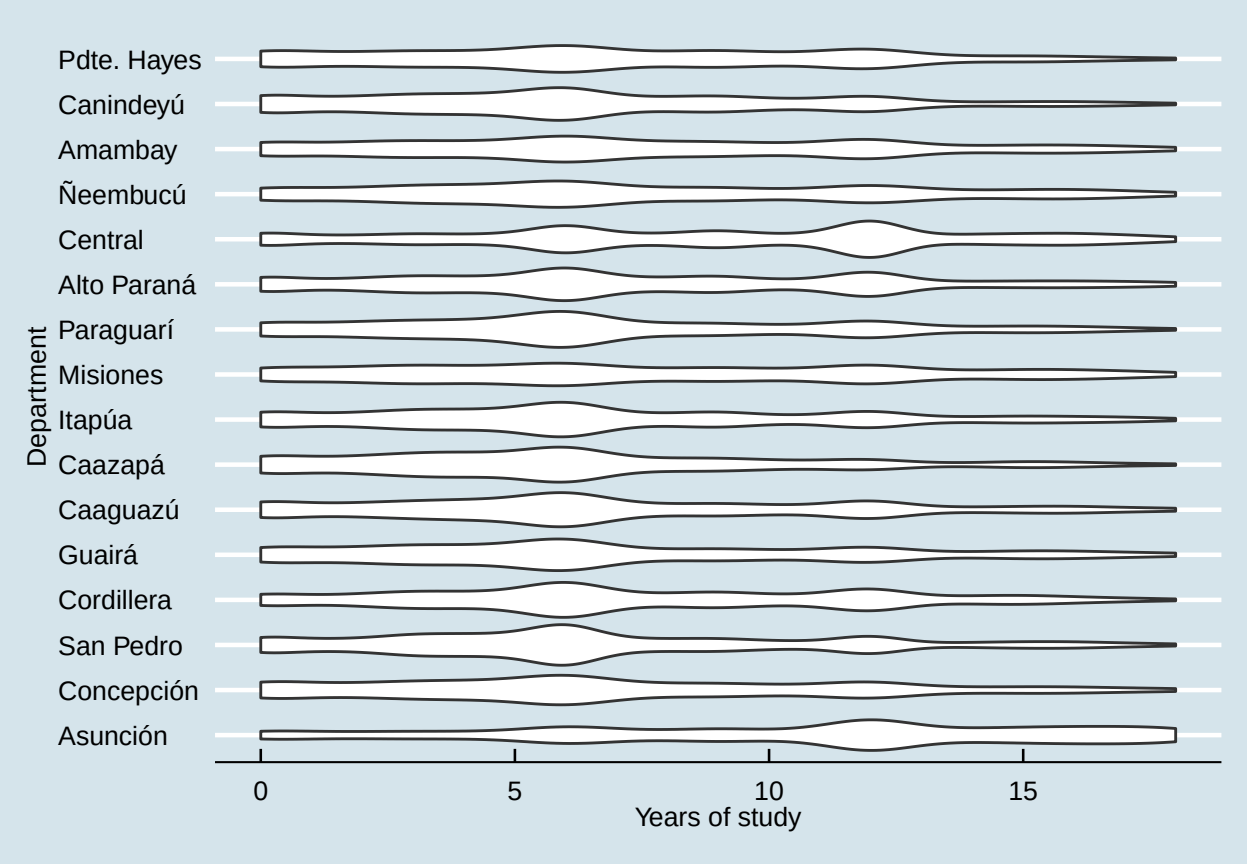
Given that difference of study years because of the language, we may wonder about the literacy rate based on the main language. The graphic below shows, as expected, that iliteracy rate is higher across the population that speaks only guarani.

```
#Literacy by language spoken
eph_train%>%
  filter(ED01%in%c("1","2","3","4"))%>%
  filter(ED02%in%c("1","6"))%>%
  ggplot(aes(x=as_factor(ED01)))+
  geom_bar(aes(fill=as_factor(ED02)))+
  ylab("Count")+
  xlab("Language spoken")+
  labs(fill="Literate")+
  theme_economist()
```

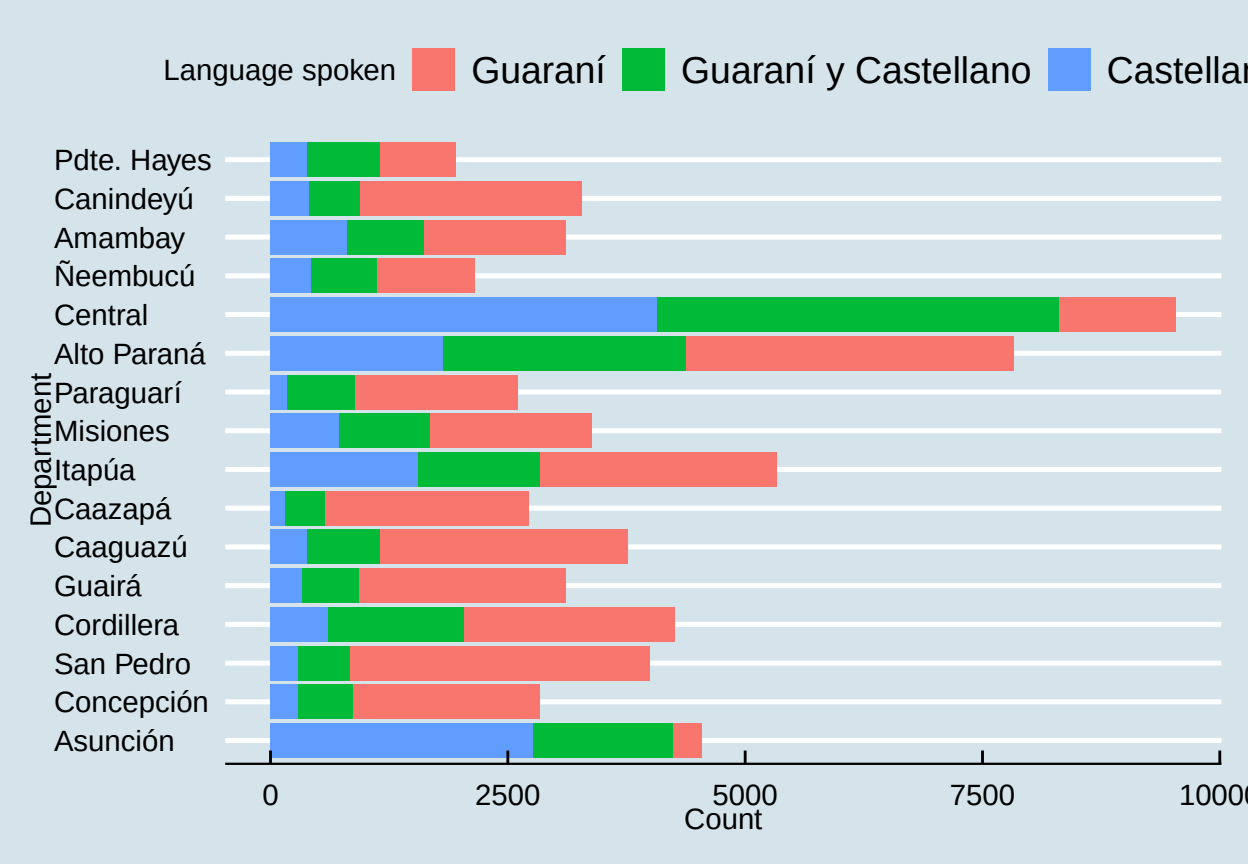
Study-related variables

We may recall a previous plot, which showed us that the higher median income was in Asuncion, where most people speaks only spanish. We can expect as well, based on the language spoken by departments, that places where spanish has high prevalence are going to have higher average years of schooling. Let us check that.

```
#Years of study by department
eph_train%>%
  filter(añoest%in%c('0','1","2","3","4",'5','6','7','8','9','10','11','12','13','14','15','16','17','18
  ggplot(aes(y=as.numeric(añoest),x=as_factor(DPTO)))+
  geom_violin()+
  coord_flip()+
  ylab("Years of study")+
  xlab("Department")+
  theme_economist()
```

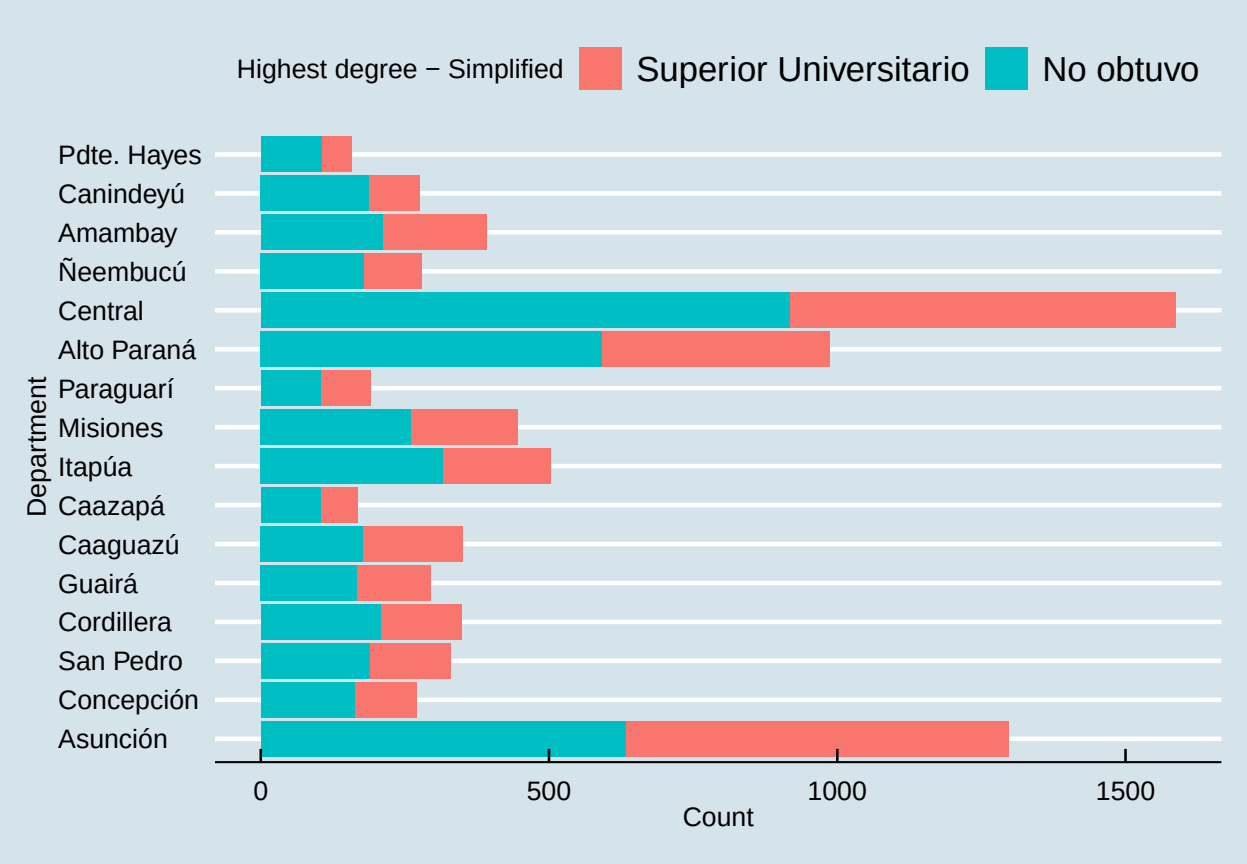



```
#Language spoken by department
eph_train%>%
  filter(ED01%in%c("1","2","3"))%>%
  ggplot(aes(x=as_factor(DPTO)))+
  geom_bar(aes(fill=as_factor(ED01)))+
  coord_flip()+
  ylab("Count")+
  xlab("Department")+
  labs(fill="Language spoken")+
  theme_economist()
```



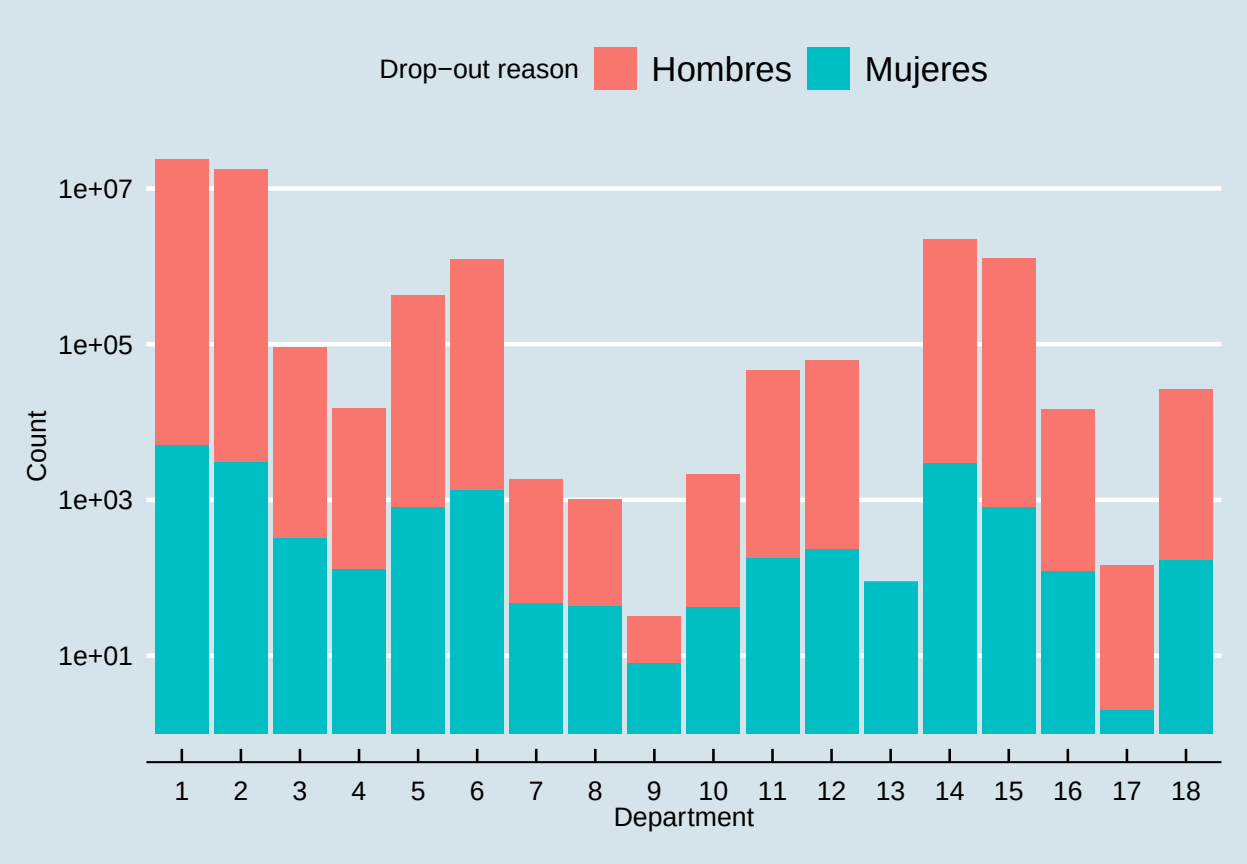
We can see in the distributions, that people of Asuncion and Central (a contiguous department) have higher chances of finishing highschool. This can be noticed in the next plot that show us that the places with the most people finishing university is in these places.

```
#Highest degree by department
eph_train%>%
  filter(ED06C%in%c("1","14"))%>%
  group_by(ED06C)%>%
  ggplot(aes(x=as_factor(DPTO)))+
  geom_bar(aes(fill=as_factor(ED06C)))+
  coord_flip()+
  ylab("Count")+
  xlab("Department")+
  labs(fill="Highest degree - Simplified")+
  theme_economist()
```



Now, we may ask, the people that dropout studies does it because of the language? Let us explore that. First, let us check the distribution of dropout reasons.

```
#Dropout by reason and sex
eph_train%>%
  filter(ED10%in%c("1","2","3","4","5","6","7","8","9","10","11","12",'13',"14","15","16","17","18"))%>%
  group_by(ED10)%>%
  ggplot(aes(x=as.factor(ED10)))+
  geom_bar(aes(fill=as_factor(P06)))+
  scale_y_log10()+
  ylab("Count")+
  xlab("Department")+
  labs(fill="Drop-out reason")+
  theme_economist()
```



First, we can see that there are 18 reasons that push people to dropout studies and in most of them, there is no difference between men and women, but in some of them there are. Let us explore that.

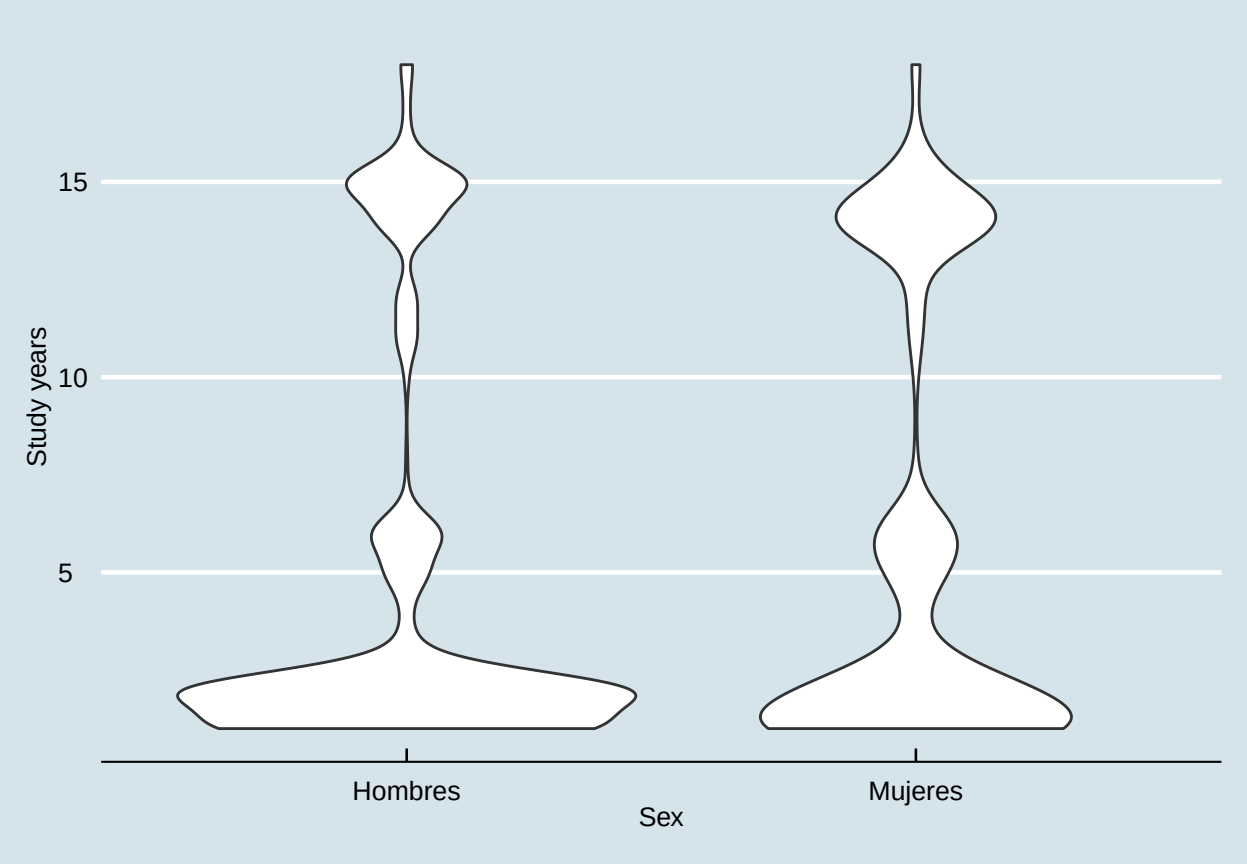
```
#Summarize dropout reasons by sex
eph_train%>%
  filter(ED10%in%c("1","2","3","4","5","6","7","8","9","10","11","12",'13',"14","15","16","17","18"))%>%
  group_by(as_factor(ED10))%>%
  summarize(male_prop=mean(P06==1), female_prop= 1-male_prop)
```

```
## # A tibble: 18 x 3
##    `as_factor(ED10)`                                male_prop female_prop
##    <fct>                                                <dbl>       <dbl>
##  1 Sin recursos en el hogar                             0.473       0.527
##  2 Necesidad de trabajar                                0.648       0.352
##  3 Muy costosos los materiales y matrículas             0.460       0.540
##  4 No tiene edad adecuada                               0.466       0.534
##  5 Considera que terminó los estudios                   0.396       0.604
##  6 No existe institución cercana                        0.401       0.599
##  7 Institución cercana muy mala                         0.442       0.558
##  8 Centro educativo cerró                               0.343       0.657
##  9 Docente no asiste con regularidad                    0.333       0.667
## 10 Institución no ofrece escolaridad completa           0.543       0.457
## 11 Requiere educación especial                          0.589       0.411
## 12 Por enfermedad                                       0.52        0.48
## 13 Realiza labores del hogar                            0.0110      0.989
## 14 Motivos familiares                                   0.197       0.803
```

```
## 15 No quiere estudiar                                          0.661       0.339
## 16 Asiste a enseñanza vocacional o formación profesional       0.492       0.508
## 17 Servicio militar                                            0.973       0.0270
## 18 Otra razón                                                  0.474       0.526
```

We may notice that military service (reason number 17) has 97% of men proportion, while others like house chores (reason number 13) have a 98% of female proportion. This lead us to the question, does your sex make you more or less likely to finish highschool? Let us explore that.

```
#Study years by sex
eph_train%>%
  filter(ED10%in%c("1","2","3","4","5","6","7","8","9","10","11","12",'13',"14","15","16","17","18"))%>%
  ggplot(aes(x=as_factor(P06), y=as.numeric(ED10)))+
  geom_violin()+
  xlab("Sex")+
  ylab("Study years")+
  theme_economist()
```

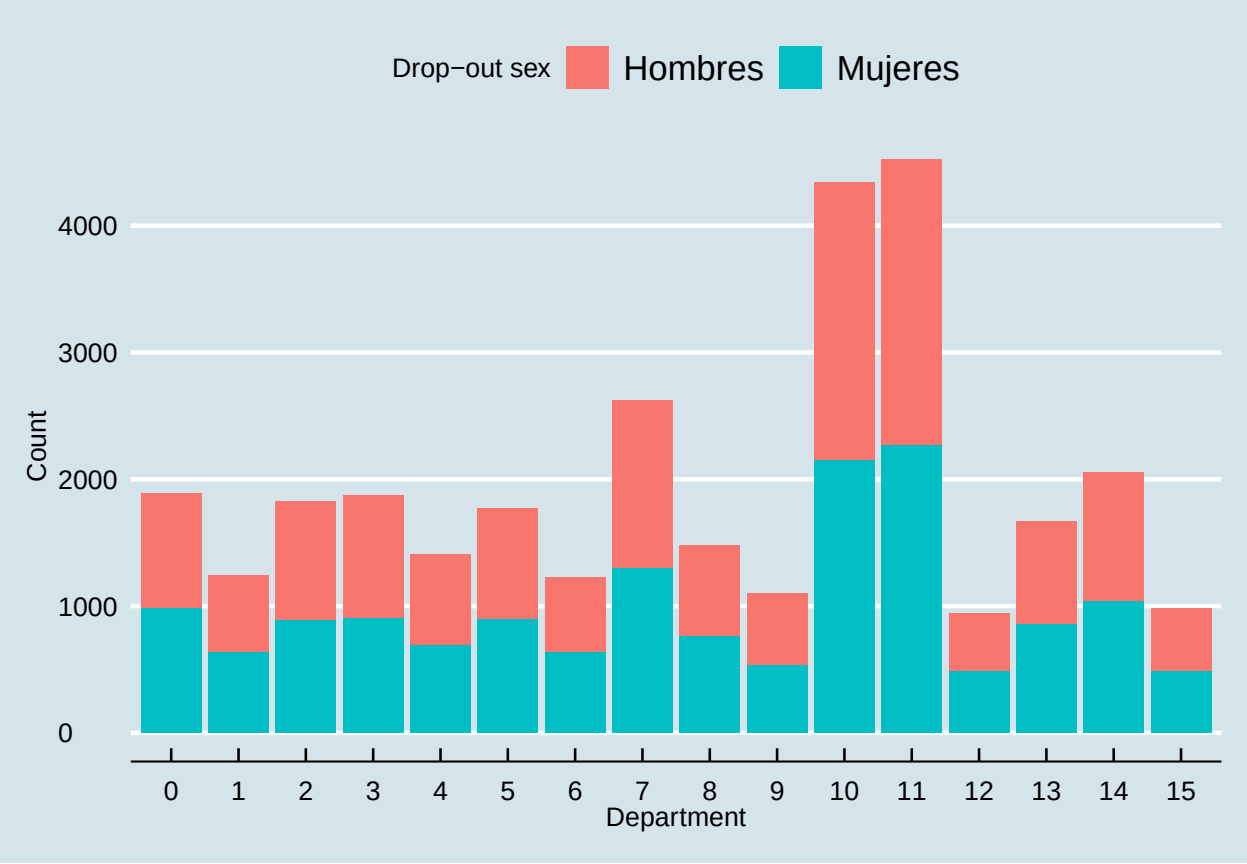


We can notice that both distributions are similarly bimodal.

Now, we can explore if there is any difference in the dropout proportion by department and sex of the people. And, again, they both look similar.

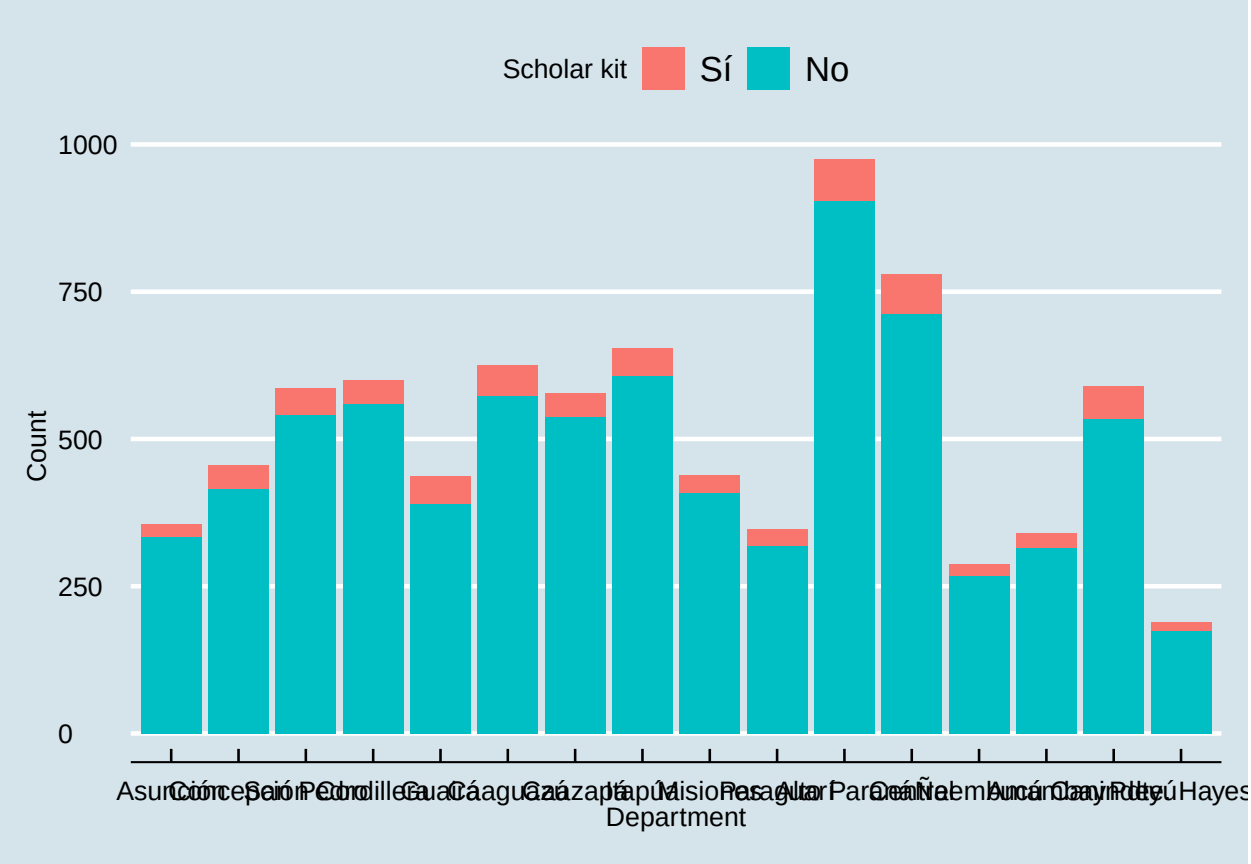
```
#Dropout by department and sex
eph_train%>%
  filter(ED10%in%c("1","2","3","4","5","6","7","8","9","10","11","12",'13',"14","15","16","17","18"))%>%
```

```
  group_by(ED10)%>%
  ggplot(aes(x=as.factor(DPTO)))+
  geom_bar(aes(fill=as_factor(P06)))+
  ylab("Count")+
  xlab("Department")+
  labs(fill="Drop-out sex")+
  theme_economist()
```



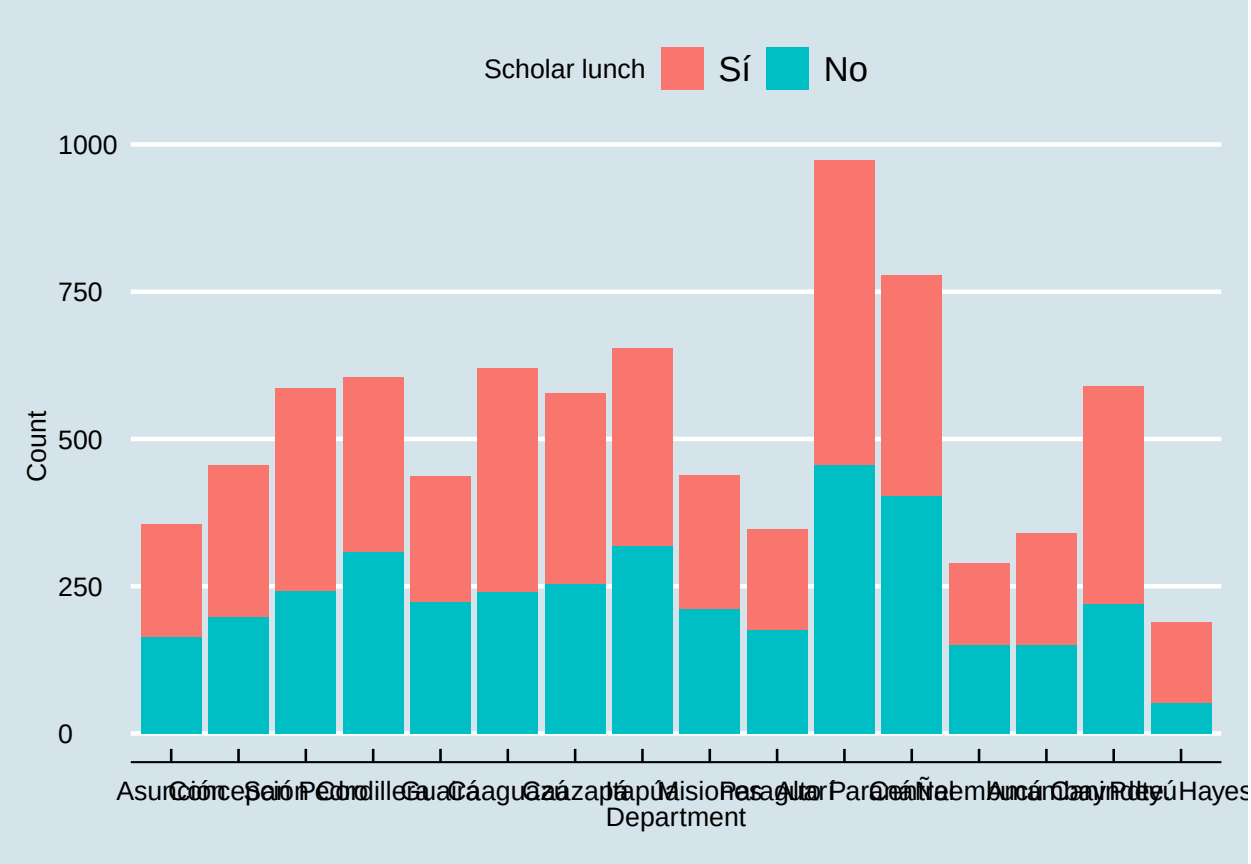
Something we did not explain before, is that a common reason to dropout highschool is not being financially able to pay for it, so that around 10 years ago the government started distributing a scholar kit with some basic materials. However, this has not reach yet the objective of benefit all students across the country. Here a plot.

```
#Scholar kit by department
eph_train%>%
  filter(ED11B9%in%c("1","6"))%>%
  group_by(ED11B9)%>%
  ggplot(aes(x=as_factor(DPTO)))+
  geom_bar(aes(fill=as_factor(ED11B9)))+
  ylab("Count")+
  xlab("Department")+
  labs(fill="Scholar kit")+
  theme_economist()
```



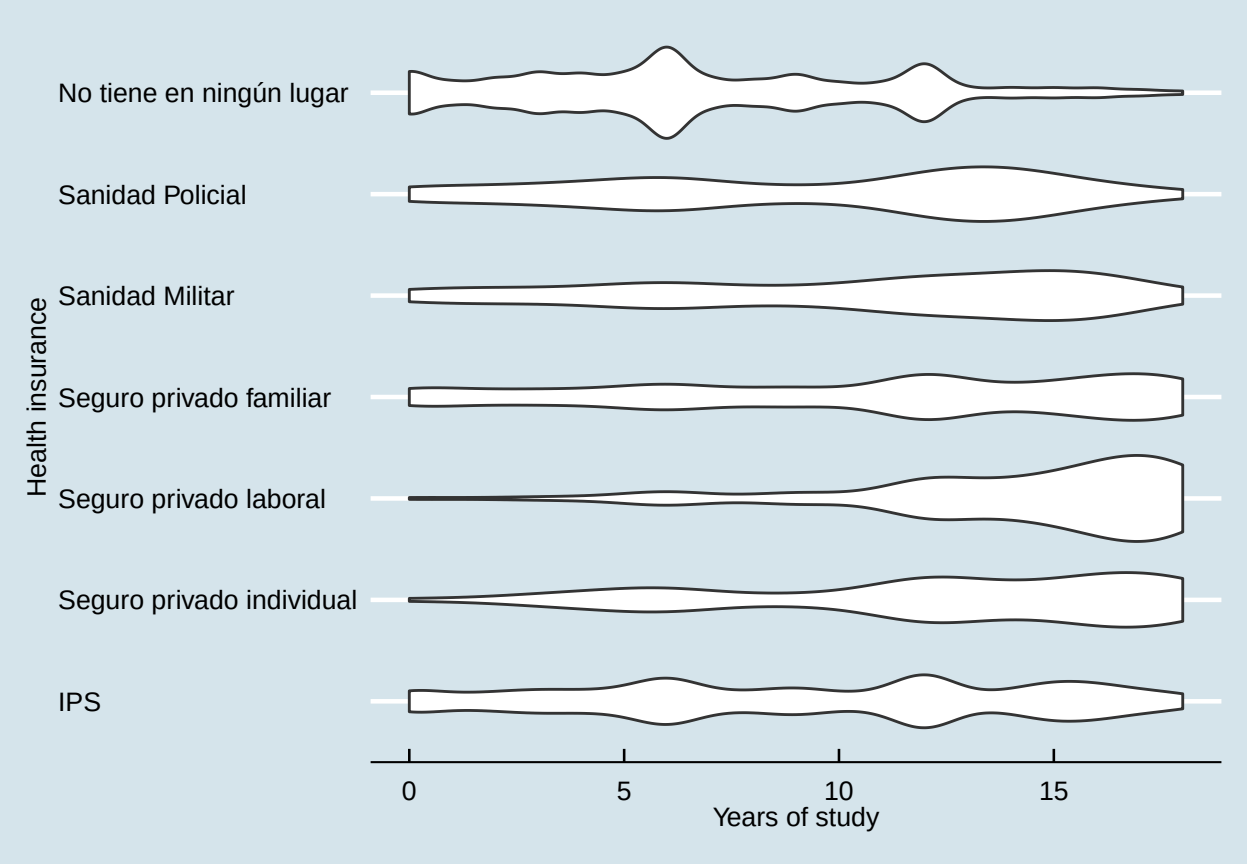
Following the same principles, the government started distributing scholar lunch as well with some improvements in benefit those who need it the most.

```
#Scholar lunch by department
eph_train%>%
  filter(ED11F1%in%c("1","6"))%>%
  group_by(ED11F1)%>%
  ggplot(aes(x=as_factor(DPTO)))+
  geom_bar(aes(fill=as_factor(ED11F1)))+
  ylab("Count")+
  xlab("Department")+
  labs(fill="Scholar lunch")+
  theme_economist()
```



We can recall that one of the reasons that force people to dropout highschool, is sickness. This is partially explained by the fact that the government is still working to strengthen the public healtcare system and cannot yet help everyone. Let us explore the distribution of years of study by kind of health insurance.

```
#Years of study by health insurance
eph_train%>%
  filter(añoest%in%c('0',"1","2","3","4",'5','6','7','8','9','10','11','12','13','14','15','16','17','18
  filter(S01A%in%c("1","2","3","4",'5','6','7'))%>%
  ggplot(aes(y=as.numeric(añoest),x=as_factor(S01A)))+
  geom_violin()+
  coord_flip()+
  ylab("Years of study")+
  xlab("Health insurance")+
  theme_economist()
```



Area and department related variables

We can recall that theres is a slight difference in the median income between urban and rural areas. It may be important to visualize the difference proportion of population by area in each department. Here a table.

```
#Summarize departments population by area
eph_train%>%
  group_by(as_factor(DPTO))%>%
  summarize(urban_prop=mean(AREA==1), rural_prop= 1-urban_prop)
```

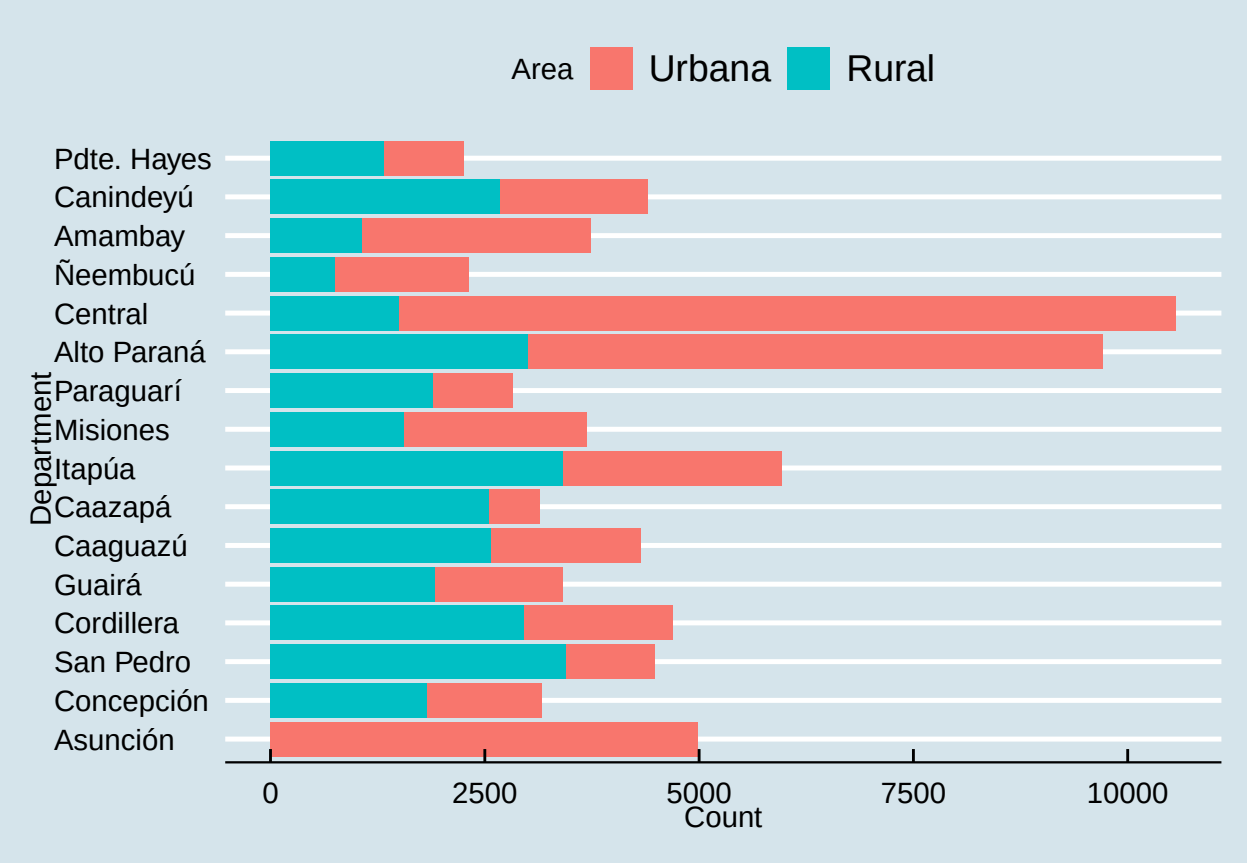
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 16 x 3
##    `as_factor(DPTO)` urban_prop rural_prop
##    <fct>                  <dbl>      <dbl>
##  1 Asunción               1          0
##  2 Concepción             0.422      0.578
##  3 San Pedro              0.230      0.770
##  4 Cordillera             0.372      0.628
##  5 Guairá                 0.436      0.564
##  6 Caaguazú               0.403      0.597
##  7 Caazapá                0.190      0.810
##  8 Itapúa                 0.427      0.573
##  9 Misiones               0.579      0.421
## 10 Paraguarí              0.329      0.671
```

```
## 11 Alto Paraná            0.691      0.309
## 12 Central                0.858      0.142
## 13 Ñeembucú               0.677      0.323
## 14 Amambay                0.714      0.286
## 15 Canindeyú              0.390      0.610
## 16 Pdte. Hayes            0.412      0.588
```

And here, a useful graphic.

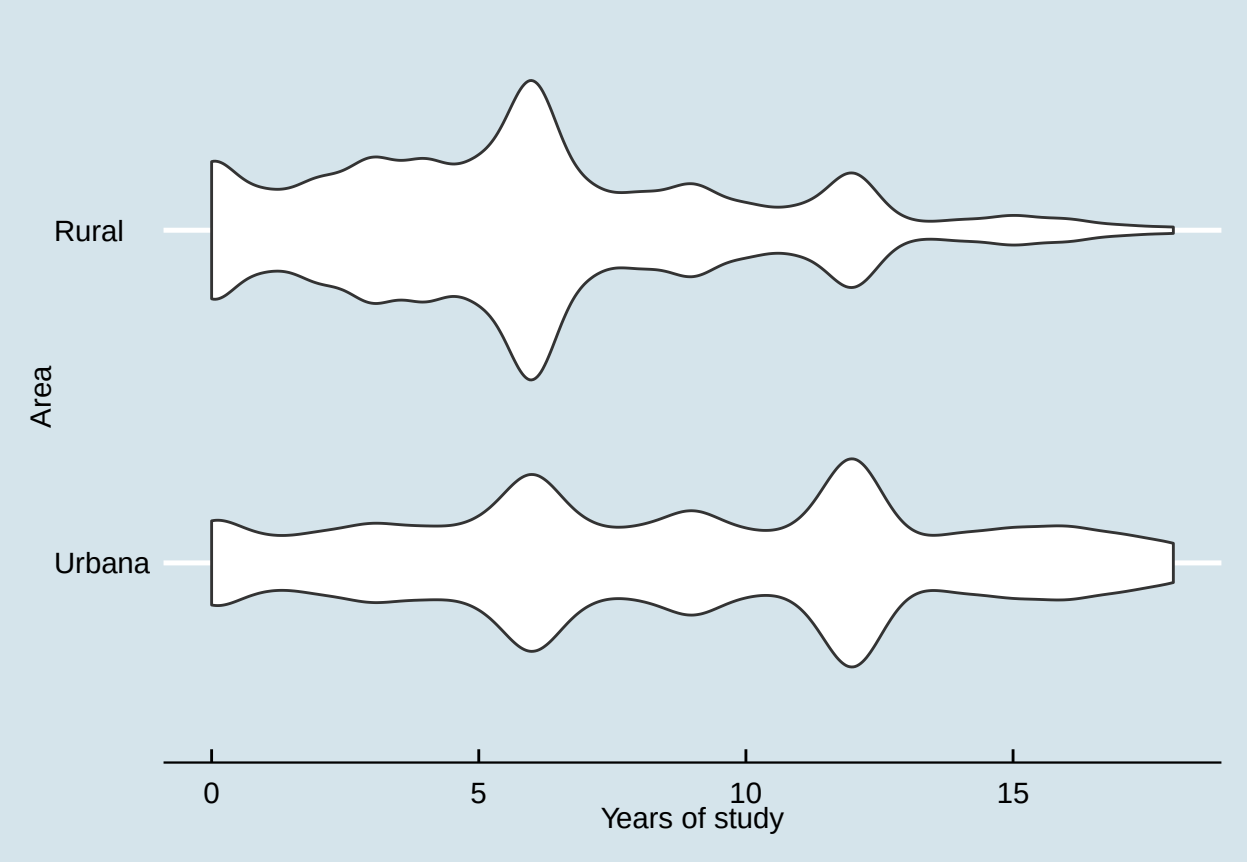
```
#Department population by area
eph_train%>%
  ggplot(aes(x=as_factor(DPTO)))+
  geom_bar(aes(fill=as_factor(AREA)))+
  coord_flip()+
  ylab("Count")+
  xlab("Department")+
  labs(fill="Area")+
  theme_economist()
```



An now, finally, before proceeding to the next step, we can plot the years of study by area.

```
#Years of study by area
eph_train%>%
  filter(añoest%in%c('0',"1","2","3","4",'5','6','7','8','9','10','11','12','13','14','15','16','17','18
  ggplot(aes(y=as.numeric(añoest),x=as_factor(AREA)))+
  geom_violin()+
```

```
  coord_flip()+
  ylab("Years of study")+
  xlab("Area")+
  theme_economist()
```



Data Cleaning

Now, as mentioned before, we need to do mainly two things with the dataset before modeling.

First, we have to create a `desertor` variable in order to have a response variable. This variable will be defined by some characteristics:

- If there is an observation in the `ED10` variables (reason to stop studying) means that the person stopped studying at some point in their life.
- Since we are talking about *highschool* dropout, we also want the person to stop study before 9th grade, this can be notice in the `ED0504` variable. This is tricky because around 20 years ago there was a reform in the educational system, so that we have to take into account both 9th grade before and after the reform.
- At last, we will split the `eph_train` dataset into `train_set` and `test_set`, in order to avoid using the `eph_test` before the final validation.

After that, we also have to turn the labelled variables into factor variables so that is easier to work with them and model an algorithm.

```
#Creating a 'desertor' variable
eph_train <- eph_train %>% filter(!is.na(ED0504))%>%filter(!is.na(decili))%>%
  mutate(desertor = as.factor(ifelse(ED0504 < 503 &ED0504 != 409 & !is.na(ED10), "desertor", "no_desert

#Transforming labelled variables into regular factor variables
eph_train <- eph_train %>% mutate_at(
  vars('P06', 'DPTO', 'AREA', 'ED01'),
  funs(as_factor(.))
)
```

```
## Warning: 'funs()' is deprecated as of dplyr 0.8.0.
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with 'tibble::lst()':
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_warnings()' to see where this warning was generated.
```

```
#Proportion of desertors
mean(eph_train$desertor=="desertor")
```

```
## [1] 0.200342
```

```
#Splitting into train and test sets
test_index <- createDataPartition(eph_train$desertor, times=1, p=0.2, list = F)
train_set <- eph_train[-test_index,]
test_set <- eph_train[test_index,]
```

```
## Warning: The 'i' argument of ''['()' can't be a matrix as of tibble 3.0.0.
## Convert to a vector.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_warnings()' to see where this warning was generated.
```

```
#Creating a 'desertor' variable
eph_test <- eph_test %>% filter(!is.na(ED0504))%>% filter(!is.na(decili))%>%
  mutate(desertor = as.factor(ifelse(as.numeric(ED0504) < 503 &ED0504 != 409 & !is.na(ED10), "desertor"

#Transforming labelled variables into regular factor variables
eph_test <- eph_test %>% mutate_at(
  vars('P06', 'DPTO', 'AREA', 'ED01'),
  funs(as_factor(.))
)
```

Modeling

When talking about a data science project, there are mainly two types of work that can be done: regression and classification. Since this project is based on a binomial classification problem, a linear model approach

may not be useful. However, we will start with a Bayes Generalized Linear Model algorithms using the `train` function of the `caret` package and then go a little bit deeper.

It is essential to point out that the `train` function first runs a prepocesing step of converting all factor variables into "dummy variables", that means turning one multivariable column into several binomial columns and then, while running the algorithm, performs cross validation (in this case we chose to run 3 cross validations) to find the best tunning parameters and gini impurity.

Once we have trained the algorithms with the `train_set`, we are going to use the `predict` function on the `test_set` and the, once achieved a good score, we will use the entire `eph_train` and `eph_test` dataset for final validation.

Variables used for modeling

From previous plots, we can recall several variables to use:

-`P02`: age of each person.

-`P06`: sex of each person.

-`DPTO`: department where the person lives.

-`AREA`: area where the person lives (urban or rural).

-`ED01`: language spoken in the house of the person.

-`decili`: decil of income to which the person belongs

-`S01A`: type of healts insurance that the person have.

Bayes GLM

The one advantage of the Bayesian GLM approach is the use of external information to improve the estimates of the linear model coefficients, so that we start with this first approach.

Naive Bayes

The naive bayes algorithm is not called in that way because it is simple or stupid, but rather because makes the strong assumption of independence across all variables. If this is true, it may perform better than other models, if not it performs very badly. However, we may recall from previous plot that variables are not independent.

Random forest

A common approach to tackle data science problems is the decision tree. But in this case we are going one step further and apply random forest. This means that instead of just one tree, our algorithm will run several trees with differents cross validations.

Results

Model Evaluation Function

For the evaluation of the model we will use the `F_meas` and `Sensitivity` functions available in the `caret` package. However, this function it is not available in the arguments of the `train` function, so that we have to create it.

```r
#Creating the metric function
f1 <- function(data, lev = NULL, model = NULL) {
  f1_val <- F1_Score(y_pred = data$pred, y_true = data$obs, positive = lev[1])
  c(F1 = f1_val)
}
```

An essential step is to remind that we are working with an imbalance problem. That means that there are more `no_desertor` than `desertor` in the dataset, so that we have to 'weight' all the models.

```r
#Creatting the weights for the imbalanced model
model_weights <- ifelse(train_set$desertor == "desertor",
                        (1/table(train_set$desertor)[1]) * 0.5,
                        (1/table(train_set$desertor)[2]) * 0.5)
sum(model_weights)#The sum MUST equals 1
```

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

Bayes GLM Evaluation

As mentioned, we are going to start with a Bayes Generalized Linear Model.

```r
#Bayesian general linear model===============
glmbayes<-train(desertor ~P02+P06+DPTO+AREA+ED01+decili+S01A,
                data = train_set,
                method="bayesglm",
                weights = model_weights,
                metric="F1",
                trControl = trainControl(summaryFunction = f1,
                                         method = "cv",
                                         number = 3,
                                         classProbs = TRUE))
glmbayes$results
```

```
##   parameter        F1        F1SD
## 1      none 0.4241589 0.001035153
```

```r
#Prediction for glmbayes
pred_glm<-predict(glmbayes, test_set)

#Confusion matrix for prediction vs observer
confusionMatrix(pred_glm, test_set$desertor)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    desertor no_desertor
##   desertor        1775        4059
##   no_desertor      826        6323
##
##                Accuracy : 0.6237
##                  95% CI : (0.6153, 0.6321)
```

```
##     No Information Rate : 0.7997
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.1988
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.6824
##             Specificity : 0.6090
##          Pos Pred Value : 0.3043
##          Neg Pred Value : 0.8845
##              Prevalence : 0.2003
##          Detection Rate : 0.1367
##    Detection Prevalence : 0.4494
##       Balanced Accuracy : 0.6457
##
##        'Positive' Class : desertor
##
```

As expected, this model does not have a very high prediction power nor a high F1 score.

```
#F1 score
F_meas(pred_glm, test_set$desertor)
```

```
## [1] 0.4208654
```

```
#Results dataframe
results<- data.frame(Method="Bayesian GLM",
                     F1_Score=F_meas(pred_glm, test_set$desertor),
                     Sensitivity=sensitivity(pred_glm, test_set$desertor))
results
```

```
##         Method  F1_Score Sensitivity
## 1 Bayesian GLM 0.4208654   0.6824298
```

Can we do it better? Yes, definitely we can.

Naive Bayes Evaluation

Now, we proceed to run a Naive Bayes model.

```
naive_bayes<-train(desertor ~ P02+P06+DPTO+AREA+ED01+decili+S01A,
                   data = train_set,
                   method="naive_bayes",
                   weights = model_weights,
                   metric="F1",
                   trControl = trainControl(summaryFunction =f1,
                                            method = "cv",
                                            number = 3,
                                            classProbs = TRUE))
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
## Warning in train.default(x, y, weights = w, ...): missing values found in
## aggregated results
```

```
naive_bayes$results
```

```
##   usekernel laplace adjust        F1        F1SD
## 1     FALSE       0      1 0.4570298 0.007668219
## 2      TRUE       0      1       NaN          NA
```

```
#Predicting with 'naive bayes' method
pred_naive_bayes<-predict(naive_bayes, test_set)

#Confusion matrix prediction vs observation
confusionMatrix(pred_naive_bayes, test_set$desertor)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    desertor no_desertor
##   desertor        1839        3699
##   no_desertor      762        6683
##
##                Accuracy : 0.6564
##                  95% CI : (0.6482, 0.6646)
##     No Information Rate : 0.7997
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2465
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7070
##             Specificity : 0.6437
##          Pos Pred Value : 0.3321
##          Neg Pred Value : 0.8976
##              Prevalence : 0.2003
##          Detection Rate : 0.1416
##    Detection Prevalence : 0.4266
##       Balanced Accuracy : 0.6754
##
##        'Positive' Class : desertor
##
```

While we can see that there was a small improvement, this may not be enough yet.

```
#F1 score
F_meas(pred_naive_bayes, test_set$desertor)
```

```
## [1] 0.4608911
```

```r
#Results dataframe
results<- bind_rows(results,data.frame(Method="Naive Bayes",
                                       F1_Score=F_meas(pred_naive_bayes, test_set$desertor),
                                       Sensitivity=sensitivity(pred_naive_bayes, test_set$desertor)))
results
```

```
##         Method  F1_Score Sensitivity
## 1 Bayesian GLM 0.4208654   0.6824298
## 2  Naive Bayes 0.4608911   0.7158785
```

Again, can we do it better? Yes, we still can.

Random Forest Evaluation

Finally, we are going to apply a random forest.

```r
#Random forest with `rpart` method=======
cart<-train(desertor ~P02+P06+DPTO+AREA+ED01+decili+S01A,
            data = train_set,
            method="rpart",
            weights = model_weights,
            metric="F1",
            trControl = trainControl(summaryFunction = f1,
                                     method = "cv",
                                     number = 3,
                                     classProbs = TRUE),
            tuneGrid= data.frame(cp=seq(0.0,0.1, length = 25)))
cart$results
```

```
##             cp        F1         F1SD
## 1  0.000000000 0.6329656 0.0004048449
## 2  0.004166667 0.6312304 0.0119579091
## 3  0.008333333 0.6319632 0.0069470616
## 4  0.012500000 0.6319632 0.0069470616
## 5  0.016666667 0.6319632 0.0069470616
## 6  0.020833333 0.5789518 0.0055208965
## 7  0.025000000 0.5789518 0.0055208965
## 8  0.029166667 0.5789518 0.0055208965
## 9  0.033333333 0.5789518 0.0055208965
## 10 0.037500000 0.5789518 0.0055208965
## 11 0.041666667 0.5789518 0.0055208965
## 12 0.045833333 0.5789518 0.0055208965
## 13 0.050000000 0.5789518 0.0055208965
## 14 0.054166667 0.5789518 0.0055208965
## 15 0.058333333 0.5789518 0.0055208965
## 16 0.062500000 0.5789518 0.0055208965
## 17 0.066666667 0.5789518 0.0055208965
## 18 0.070833333 0.5789518 0.0055208965
## 19 0.075000000 0.5366481 0.0349780441
## 20 0.079166667 0.5165078 0.0029415836
## 21 0.083333333 0.5165078 0.0029415836
## 22 0.087500000 0.5165078 0.0029415836
```

```
## 23 0.091666667 0.5165078 0.0029415836
## 24 0.095833333 0.5165078 0.0029415836
## 25 0.100000000 0.5165078 0.0029415836
```

```
#Prediction with the 'rpart' randomforest method
pred_cart<- predict(cart, test_set)

#Confusion martix prediction vs observation
confusionMatrix(pred_cart, test_set$desertor, mode="prec_recall")
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    desertor no_desertor
##   desertor        2173        2042
##   no_desertor      428        8340
##
##                Accuracy : 0.8098
##                  95% CI : (0.8029, 0.8165)
##     No Information Rate : 0.7997
##     P-Value [Acc > NIR] : 0.002003
##
##                   Kappa : 0.5183
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##               Precision : 0.5155
##                  Recall : 0.8354
##                      F1 : 0.6376
##              Prevalence : 0.2003
##          Detection Rate : 0.1674
##    Detection Prevalence : 0.3247
##       Balanced Accuracy : 0.8194
##
##        'Positive' Class : desertor
##
```

We can notice a great improvement!

```
#F1 score
F_meas(pred_cart, test_set$desertor)
```

```
## [1] 0.6305332
```

```
#Results dataframe
results<- bind_rows(results,data.frame(Method="Rpart",
                                       F1_Score=F_meas(pred_cart, test_set$desertor),
                                       Sensitivity=sensitivity(pred_cart, test_set$desertor)))
results
```

```
##         Method  F1_Score Sensitivity
## 1 Bayesian GLM 0.4208654   0.6824298
## 2  Naive Bayes 0.4608911   0.7158785
## 3        Rpart 0.6305332   0.8296809
```

Now, we are ready to run the final validation.

Final Validation

As previous mentioned, now we are going to perform the final validation, training the algorithm with the entire `eph_train` dataset and testing it with the `eph_test` dataset. The algorithm used will be the random forest with the `rpart` method.

```
#Final Validation
#Creatting the weights for the imbalanced final model
model_weights <- ifelse(eph_train$desertor == "desertor",
                        (1/table(eph_train$desertor)[1]) * 0.5,
                        (1/table(eph_train$desertor)[2]) * 0.5)
sum(model_weights)#The sum MUST equals 1
```

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

First, we re-run the weights for the model.

```
#Training the model
validation<-train(desertor ~P02+P06+DPTO+AREA+ED01+decili+S01A,
            data = eph_train,
            method="rpart",
            weights = model_weights,
            metric="F1",
            trControl = trainControl(summaryFunction = f1,
                                     method = "cv",
                                     number = 3,
                                     classProbs = TRUE),
            tuneGrid= data.frame(cp=seq(0.0,0.1, length = 25)))
validation$results
```

```
##             cp        F1        F1SD
## 1  0.000000000 0.6319485 0.006681624
## 2  0.004166667 0.6233618 0.004662800
## 3  0.008333333 0.6326847 0.003437630
## 4  0.012500000 0.6326847 0.003437630
## 5  0.016666667 0.6326847 0.003437630
## 6  0.020833333 0.5784915 0.004661413
## 7  0.025000000 0.5784915 0.004661413
## 8  0.029166667 0.5784915 0.004661413
## 9  0.033333333 0.5784915 0.004661413
## 10 0.037500000 0.5784915 0.004661413
## 11 0.041666667 0.5784915 0.004661413
## 12 0.045833333 0.5784915 0.004661413
## 13 0.050000000 0.5784915 0.004661413
## 14 0.054166667 0.5784915 0.004661413
## 15 0.058333333 0.5784915 0.004661413
## 16 0.062500000 0.5784915 0.004661413
## 17 0.066666667 0.5784915 0.004661413
## 18 0.070833333 0.5566234 0.033616975
## 19 0.075000000 0.5162042 0.002016371
```

```
## 20 0.079166667 0.5162042 0.002016371
## 21 0.083333333 0.5162042 0.002016371
## 22 0.087500000 0.5162042 0.002016371
## 23 0.091666667 0.5162042 0.002016371
## 24 0.095833333 0.5162042 0.002016371
## 25 0.100000000 0.5162042 0.002016371
```

```
#Prediction with the 'rpart' randomforest method
pred_valid<- predict(validation, eph_test)

#Confusion martix prediction vs observation
confusionMatrix(pred_valid, eph_test$desertor, mode="prec_recall")
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    desertor no_desertor
##   desertor        7726        7829
##   no_desertor     1547       32094
##
##                Accuracy : 0.8094
##                  95% CI : (0.8059, 0.8129)
##     No Information Rate : 0.8115
##     P-Value [Acc > NIR] : 0.8835
##
##                   Kappa : 0.5056
##
##  Mcnemar's Test P-Value : <2e-16
##
##               Precision : 0.4967
##                  Recall : 0.8332
##                      F1 : 0.6224
##              Prevalence : 0.1885
##          Detection Rate : 0.1570
##    Detection Prevalence : 0.3162
##       Balanced Accuracy : 0.8185
##
##        'Positive' Class : desertor
##
```

We can appreciate that we got a very similar result, that means a good `sensitivity` and `F1_score`

```
#F1 score
F_meas(pred_valid, eph_test$desertor)
```

```
## [1] 0.6223618
```

```
#Results dataframe of final validation
results<- bind_rows(results,data.frame(Method="Final validation - rpart",
                                       F1_Score=F_meas(pred_valid, eph_test$desertor),
                                       Sensitivity=sensitivity(pred_valid, eph_test$desertor)))
results
```

```
##                       Method  F1_Score Sensitivity
## 1              Bayesian GLM 0.4208654   0.6824298
## 2               Naive Bayes 0.4608911   0.7158785
## 3                     Rpart 0.6305332   0.8296809
## 4 Final validation - rpart 0.6223618   0.8331716
```

Conclusion

At first, we loaded the *EPH* datasets from 2017 and 2018. Used the 2017's dataset for training and the 2018's for validation. We started exploring the variables of the 2017's poll, finding insights and correlations.

After finishing our exploration, we proceed to create weights for this imbalanced problem and started a very simple algorithm as a generalized linear modeand then moving on until achieving a random forest algorithm.

Limitations

As may have been noticed, the entire dataset is made up of more than 200 variables but we ended up using less than 10 variables. This is mainly due to the computation power require to build a more robust model with more variables.

Another aspect we have not mentioned yet is that the entire *EPH* is made up of 13 datasets. In this report we used only the dataset number 2.

Future Work

As mentioned, in the future more models can be build with more variables and relating at leats one more out of the 13 datasets.

Following the same idea, another variable we did not mention is the `FEX` variables, also known as "expansion factor". This could be added to get the approcimate numbers across the entire country.