SURVIVAL RATE ANALYSIS

MSc. DATA SCIENCE AND ECONOMICS

Gregorio Luigi Saporito
Department of Economics, Management and Quantitative Methods
Department of Computer Science
University of Milan
Milan, Italy

Tommaso Pessina
Department of Economics, Management and Quantitative Methods
Department of Computer Science
University of Milan
Milan, Italy

November 6, 2020

ABSTRACT

The purpose of this analysis is to determine whether a person will conclude the subscription process to a virtual piggy bank application and/or where the client is stuck and try to infer something from the data. We will try to study the percentage of people that will really conclude the subscription process (with respect to those who are stuck in the process) and try to build a model to predict if a client will conclude or not the subscription (based on his/her behavior). The data is pre-anonymized by the company that provided them.

Keywords R · Statistics · Survival Analysis · Logistic Regression · More

1 Introduction

The original dataset provided by the company is organised as follow:

- ID: unique identifier of the event;
- COMPLETED_STEP: identifies the [completed] step of the event in JSON format;
- DATE_EVENT: it contains the date and the time of the event;
- NETWORK_ID: represents the network tracking the sequence of events that occurred;
- USER_ID: unique identifier of the user;
- SESSO: represent the gender of the user;
- ETA: indicate the age of the user;
- TEMP_PROMOTION_CODE_ID: it contains (if specified) the promotion code of the user;
- LAST_ITEM: last access to the system of the user;

Note that we talk about event because each subscription step will trigger an event on the database.
Each step represents a different passage through the subscription process and they are:

- anagrafica-a, anagrafica-b, anagrafica-c, anagrafica-d: these are the steps in which the user's personal data are requested;

- codice fiscale: here the user should provide his/her fiscal code;
- antiriciclaggio: this step refers to the anti-money laundering where the user is asked whether he/she is a politician or an "exposed" person in terms of money source;
- conclude: if a user arrives here he/she has concluded the subscription process;
- contract-activation, contract-subscription.

Starting from this dataset one can think about which kind of information it can deliver, but to see it we should run some analyses over it. As matter of fact we have grouped users by their id in order to perform the subsequent data cleaning operations.
In particular, we created a new dataset which includes:

- CLIENT_id: it is the unique identifier of the user;
- surv_time: we calculate the difference between the DATE_EVENT of the first step of a certain user and the DATE_EVENT of his last step in the process, using hours as a unit of measure;
- status: is a flag that will assume value 1 if a user arrived to the step "conclude", 0 otherwise;
- extra_attempts: indicates if the user has made extra subscription attempts in that specific part of the process;
- main_extra_attempt: it will assume a value different from "none" if the previous item is different from zero and it represents the last step in which he/she is stuck;
- stuck: reports the step (of the subscription process) in which the user is stuck;
- mean_time: it is the mean time that the user spent in each of the steps that he/she has concluded. It is calculated by diving the total time the he/she spent in the process by the number of steps (note that he can go back, restart, go forward, etc.);
- gender: contains the gender of the user (can be not specified);
- age: indicate the age of the user (can be not specified);
- prom_code_id: is the unique id of the promotion (can be not specified if the user does not insert it);
- extra_attempts_dummy: a dummy variable that assume value 1 if the user made an extra attempt (0 otherwise)
- prom_code: a dummy variable that assume value 1 if the user use a promo code, 0 otherwise;
- age_group: we divide the individual basing of their age like:
  - less than 20;
  - between 20 and 40;
  - between 20 and 60;
  - more than 60;
  - not specified.

All our analyses are based on the cleaned dataset.

2 Survival analysis

In this chapter we will discuss about the survival analysis with the aim to see how many people go all the way through the subscription process (along a time scale) or where they are stuck. Having said that, we can also see if a person will conclude the subscription after begin stuck for a while.
First of all, we see that there are many outliers for survival time and few clients go through the system for a long time. After that, we can visually analyze Figure 1 where the clients are stuck in the subscription process and other interesting breakouts.

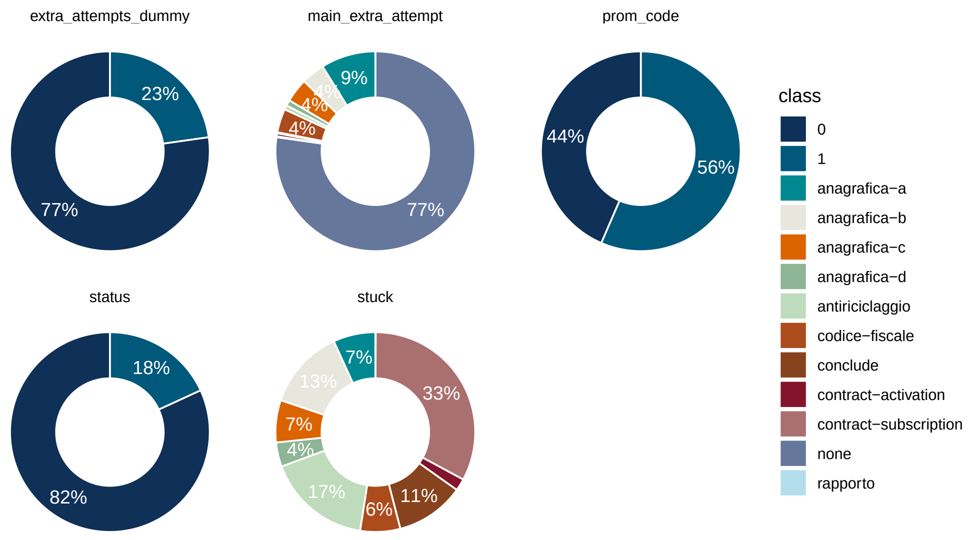

Figure 1: Donut graph of stuck users

Before discussing in detail our survival analysis, we should say a few words about the method we used. We used the "Kaplan–Meier estimator" that is a method used (especially in medical research) to estimate the probability that someone will survive over time. Here, by surviving, we mean that a user still hasn't finished the whole subscription process (i.e. he/she has not already concluded the process).
This is the basis of our analysis, that we will discuss in the next subchapter.

2.1 Kaplan-Meier estimates of the probability of survival over time

The first thing that we can say is that, as shown in Figure 2, as time passes the probability of concluding increases (i.e. the survival rate decreases).

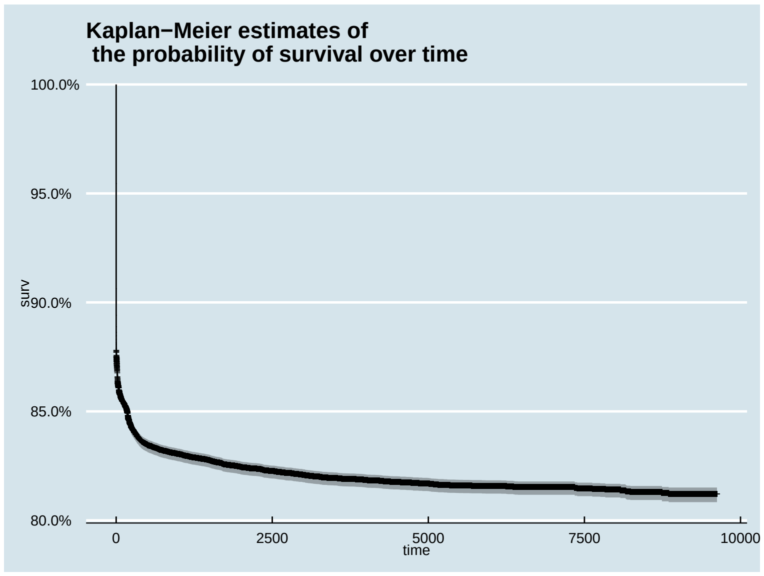

Figure 2: Probability of survival over time

Then in Figure 3 we can see that who does not provide a promotion code has a lower probability of survival

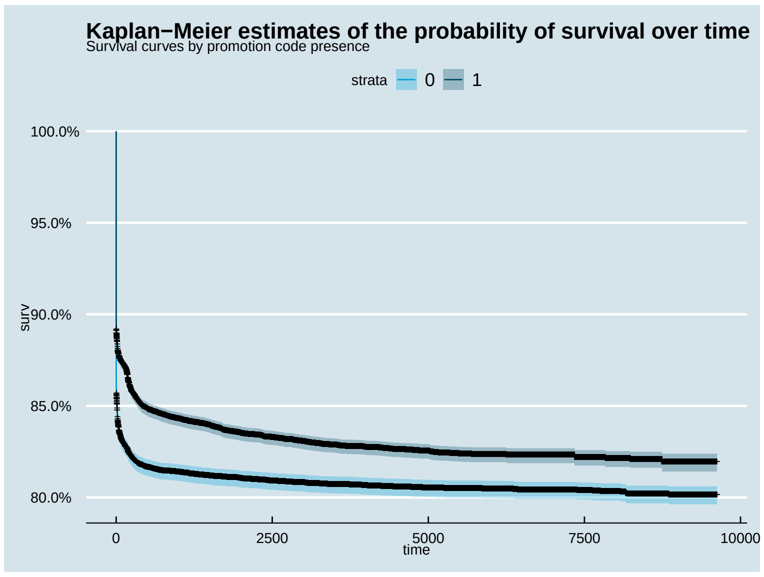

Figure 3: Probability of survival over time by promotion code

Moreover, in Figure 4 we can see that Males has a lower probability of survival over times.

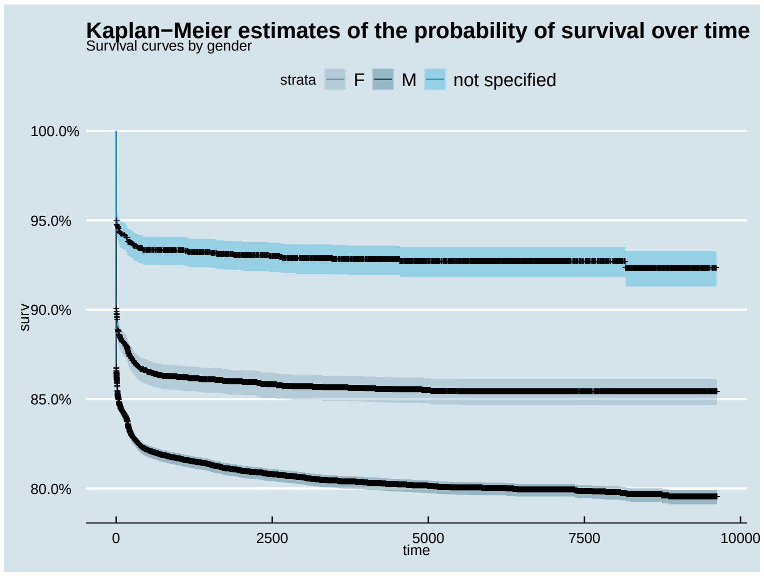

Figure 4: Probability of survival over time by gender

Now, we can look at the probability of survival over time by age group. A shown in Figure 5, the group "less than 20" has lower survival probability over time.

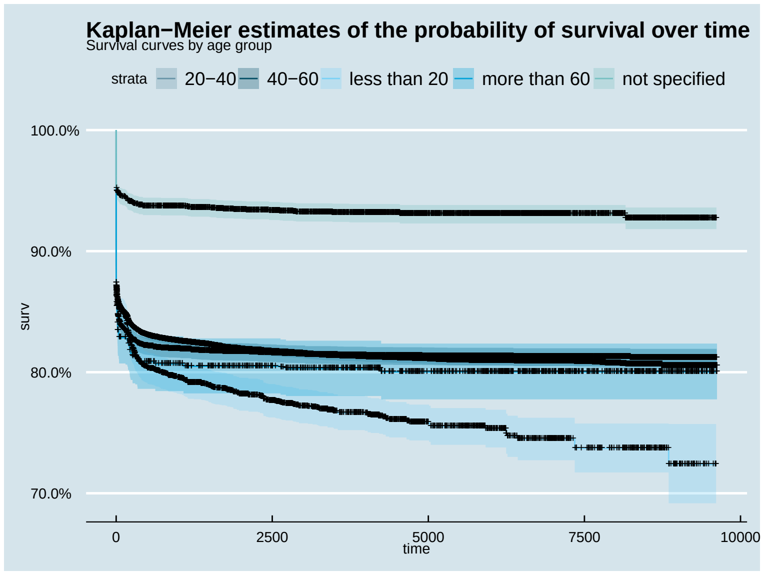

Figure 5: Probability of survival over time by age group

Having said that, we can now look at the survival curve between who makes extra attempts and who does not. As shown in Figure 6, who makes extra attempts is more likely not to conclude (since the survival rate is lower) and there are small overlaps between the curves (i.e. the confidence intervals are relatively narrow).

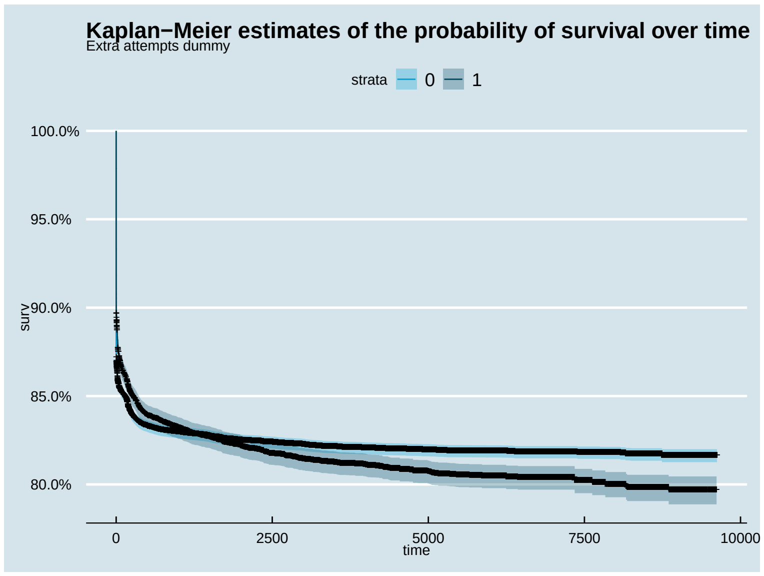

Figure 6: Probability of survival over time with extra attempt

If we look at Figure 7, we can see that who does not perform any extra attempt has a lower probability of surviving with respect to time (i.e. it is more likely that it will conclude the process sooner rather than later). Anyone who makes an extra attempt is more likely to survive, but after 7500 hours we can see that there are some overlaps.

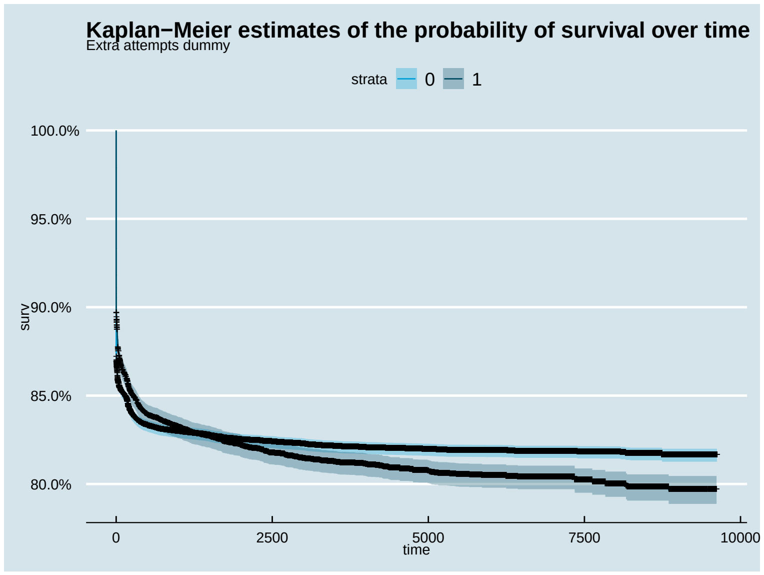

Figure 7: Probability of survival over time where main extra attempts occur

3 Predict the probability of conclude the subscription

We can now expand our analysis by creating a model for predicting the probability that a user will conclude the subscription process based on his/her behavior.
Firstly, we use the Logistic Regression model because it is particularly suited to classify dichotomous categorical variables (i.e. concluding or not concluding the process). We start by dividing the dataset into train and test (with a 70% 30% rule respectively) and with divide this data by status (which can assume value 0 or 1).
After having created these two sub-dataset (i.e. the train and the test set), we can define our model with "status" as a response variable and "stuck", "surv_time", "extra_attempts", "main_extra_attempt" and "mean_time" as explanatory

variables.
We can see below the summary of our model.

Listing 1: R output of the logist regression

```
Call:
glm(formula = status ~ stuck + surv_time + extra_attempts +
    main_extra_attempt +
    mean_time + gender + prom_code,
    family = binomial(link = "logit"),
    data = trainingData)

Deviance Residuals:
    Min        1Q    Median       3Q       Max
-4.8844   -0.0036   0.0000   0.0000   4.5176

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)              -1.444e+00  6.034e-01  -2.393 0.016724 *
stuckanagrafica-b        -6.379e-01  8.400e-01  -0.759 0.447580
stuckanagrafica-c         1.961e+00  6.729e-01   2.914 0.003565 **
stuckanagrafica-d         1.788e+00  7.114e-01   2.513 0.011979 *
stuckantiriciclaggio      1.907e+00  6.052e-01   3.152 0.001624 **
stuckcodice-fiscale       2.684e-02  7.713e-01   0.035 0.972241
stuckconclude             3.050e+01  3.262e+02   0.093 0.925518
stuckcontract-act.        1.906e+00  9.031e-01   2.110 0.034828 *
stuckcontract-sub.        5.269e+00  5.859e-01   8.993  < 2e-16 ***
stuckrapporto             3.459e+00  1.847e+00   1.873 0.061054 .
surv_time                -2.544e-03  8.413e-05 -30.233  < 2e-16 ***
extra_attempts            4.584e-02  1.130e-02   4.058 4.94e-05 ***
main_extra_anagraf-b     -1.042e+00  3.650e-01  -2.856 0.004287 **
main_extra_anagraf-c     -8.977e-01  3.926e-01  -2.287 0.022211 *
main_extra_anagraf-d     -1.127e+00  6.065e-01  -1.858 0.063198 .
main_extra_antiricicl.   -9.132e-01  6.716e-01  -1.360 0.173922
main_extra_codice-fisc.  -3.632e-01  4.264e-01  -0.852 0.394369
main_extra_attemptconc.  -4.699e+00  2.969e+03  -0.002 0.998737
main_extra_contract-act.-2.661e+00  1.138e+00  -2.338 0.019410 *
main_extra_contract-sub. 5.088e+00  1.006e+01   0.506 0.613106
main_extra_attemptnone   -1.276e+00  2.526e-01  -5.051 4.39e-07 ***
main_extra_attemptrapp.  6.623e+00  8.569e+01   0.077 0.938393
mean_time                 9.485e-03  5.627e-04  16.857  < 2e-16 ***
genderM                   6.587e-01  1.569e-01   4.197 2.70e-05 ***
gendernot specified       3.223e-01  4.102e-01   0.786 0.431959
prom_code                 4.378e-01  1.230e-01  -3.559 0.000372 ***
---
Signif. codes:  0 "***" 0.001 "**" 0.01 "*" 0.05 "." 0.1 " " 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 22793.5  on 16441  degrees of freedom
Residual deviance:  2105.1  on 16416  degrees of freedom
AIC: 2157.1

Number of Fisher Scoring iterations: 20
```

As we can see, not all the variables are significant for our estimate.
We can then compute the optimal score that minimizes the misclassification error resulting from the model above, that is 0.97.

Like in the case of linear regression, we should check for multicollinearity in the model. As seen in Table 1 below, all variables in the model have VIF below 4.

Table 1: VIF for the model

	GVIF	Df	$GVIF^(1/(2 * Df))$
stuck	3.181705	9	1.066413
surv_time	1.947976	1	1.395699
extra_attempts	1.458877	1	1.207840
main_extra_attempt	4.198197	10	1.074368
mean_time	2.047091	1	1.430766
gender	1.132867	2	1.031679
prom_code	1.076693	1	1.037638

Then, we can analyze the misclassification error, that is the percentage mismatch of predicted vs actuals, irrespective of being 1's or 0's. The lower the misclassification error, the better is the model. Given our optimal cutoff value, we obtain a misclassification error of 0.0095.
Finally, we can plot the Receiver Operating Characteristics Curve, that traces the percentage of true positives accurately predicted by a given logit model as the prediction probability cutoff is lowered from 1 to 0. For a good model, as the cutoff is lowered, it should mark more of actual 1's as positives and lesser of actual 0's as 1's. The greater the area under the ROC curve, the better the predictive ability of the model. We can see this plot in Figure 8.

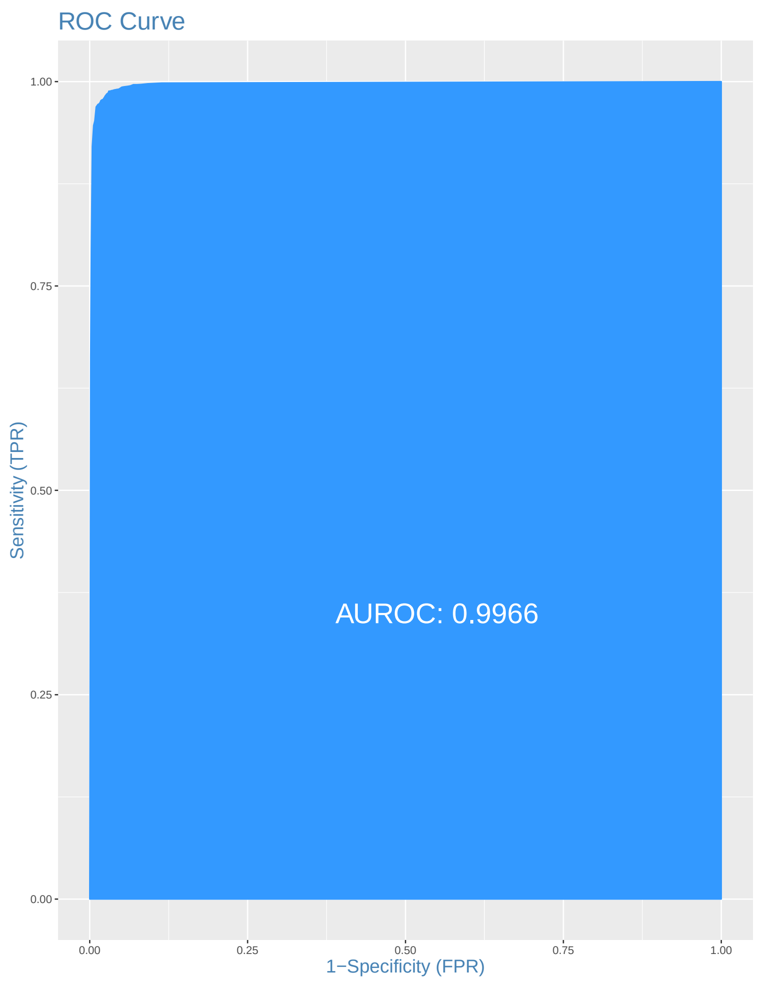

Figure 8: Receiver Operating Characteristics Curve

4 Conclusion

We can analyze the concordance of our estimation which means that the model-calculated probability-scores of all actual Positive's, (i.e. Ones) should be greater than the model-calculated probability-scores of ALL the Negatives (i.e. Zeroes). Such a model is said to be perfectly concordant and a highly reliable one. In simpler words, of all combinations of 1-0 pairs (actuals), Concordance is the percentage of pairs, whose scores of actual positive's are greater than the scores of actual negative's. For a perfect model, this will be 100%. So, the higher the concordance, the better is the quality of model.
Our model gives the following results:

- Concordance: 0.9975145, i.e 99%;
- Discordance: 0.002485465;
- 157790624 number of pairs.

We can also analyze the sensitivity and the specificity of our model. Sensitivity (or True Positive Rate) is the percentage of 1's correctly predicted by the model, while, specificity is the percentage of 0's correctly predicted. We obtain a:

- Sensitivity of 0.9205448, i.e. 92%;
- Specificity of 0.9959577, i.e. around 100%;

Despite the lack of other interesting explanatory variable such as gender and age the model still provides a good accuracy and is useful to understand which explanatory variables are significant and more relevant to explain the final outcome.
To summarize, we have analyzed the survival rate of a customer going through the subscription process and how this behavior differs based on different categories the client belongs to (i.e. extra attempts vs no extra attempts). Special attention should be payed to the segment of women because they are less likely to conclude. While there might be some intrinsic reason for this or peculiarities in the subscription process, it is worth proposing some incentives/support to minimise the risk of loosing this important customer segment.
Finally, we have fit a logistic regression to model the probability that a client concludes the process based on a set of observed predictors.

5 Code appendix

5.1 Clean data

```
library(readr)
library(tidyverse)

API_Subcription <- read_csv("API_Subcription.csv") %>%
  # remove column not needed
  select(-NETWORK_ID) %>%
  # remove part of string not needed from API
  mutate(COMPLETED_STEP = str_remove_all(COMPLETED_STEP,
                                          '/api/subscribe-|/api//subscribe-|.json')) %>%
  # we want to track if the user goes back and forth counting the number of attempts
  group_by(USER_ID,COMPLETED_STEP) %>%
  arrange(USER_ID,COMPLETED_STEP,DATE_EVENT, ID) %>%
  mutate(attempt = 1:n()) %>%
  ungroup() %>%
  rename(GENDER = SESSO, AGE = ETA)

# get ids of people
ids <- API_Subcription %>%
  pull(USER_ID) %>% unique()

# some users just have one conclude without the previous steps


# function to calculate the survival time
getsurvtime <- function(){
  if('conclude' %in% x$COMPLETED_STEP){
    final <- max(filter(x,COMPLETED_STEP=="conclude")$DATE_EVENT)
    initial <- min(x$DATE_EVENT)
    difftime(final, initial, units = "hours")
  }else{
    # the last time the dataset was extracted
    final <- as.POSIXct("2020-11-04␣23:35:00", tz = "UCT")
    initial <- min(x$DATE_EVENT)
    difftime(final, initial, units = "hours")
  }
}

# we need to count for how long each client has survived
data <- tibble()
count_ids = 0
perc = 0

for(i in ids) {
  x <- API_Subcription %>%
    filter(USER_ID == i)

  # survival time in hours
  time_diff <- getsurvtime()

  mean_time <- difftime(max(x$DATE_EVENT), min(x$DATE_EVENT),
                        units = "hours")/nrow(x)

  extra_attempts <- 0
  for(s in x$COMPLETED_STEP){
    ctesp <- x %>%
      filter(COMPLETED_STEP == s) %>%
      nrow()
    extra_attempts <- extra_attempts + ctesp - 1
  }

  # status 1 if death occurred (conclude)

  status <- ifelse(nrow(filter(x,COMPLETED_STEP=='conclude'))==0,0,1)

  main_extra_attempt <- filter(filter(x, attempt > 1),
                               attempt==max(attempt))$COMPLETED_STEP[1]

  stuck <- x %>%
    filter(DATE_EVENT==max(DATE_EVENT)) %>%
    pull(COMPLETED_STEP)

  prom <- x$TEMP_PROMOTION_CODE_ID[1]

  data <- data %>%
    bind_rows(
      tibble(
        CLIENT_id = i,
        # time in seconds
        surv_time = as.double(time_diff),
        status = status,
        extra_attempts = extra_attempts,
        main_extra_attempt = main_extra_attempt,
        stuck = stuck,
        mean_time = mean_time,
        gender = x$GENDER[1],
        age = x$AGE[1],
        prom_code_id = prom
      )
    )

  # track percentage until complete
  count_ids <- count_ids + 1

  if(perc!=round((count_ids/length(ids))*100)){
    print(paste0("complete",round((count_ids/length(ids))*100),"%"))
  }
  perc <- round((count_ids/length(ids))*100)
}

write_csv(data,"data.csv")
```

5.2 Main

```
library(readr)
library(tidyverse)
library(survival)
library(ggfortify)
library(ggthemes)

data <- read_csv("data.csv") %>%
  # extra attempts dummy
  mutate(extra_attempts_dummy = ifelse(extra_attempts==0,0,1)) %>%
  mutate(main_extra_attempt = ifelse(is.na(main_extra_attempt),"none",
                                     main_extra_attempt)) %>%
  mutate(main_extra_attempt = factor(main_extra_attempt)) %>%
  mutate(prom_code = ifelse(is.na(prom_code_id),0,1)) %>%
  mutate(gender = ifelse(is.na(gender),"not␣specified", gender)) %>%
  mutate(age_group = case_when(
    age >= 0 & age < 20   ~ "less␣than␣20",
    age >= 20 & age < 40  ~ "20-40",
    age >= 40 & age < 60  ~ "40-60",
    age >= 60  ~ "more␣than␣60",
    is.na(age) ~ "not␣specified"
  )) %>%
  mutate(stuck = factor(stuck),
         gender = factor(gender),
         age_group = factor(age_group))

# boxplot of survival time
boxplot(data$surv_time)

# donut charts to explore the data
type <- c("status","extra_attempts_dummy","main_extra_attempt","stuck","prom_code")

donut_df <- map_df(type, function(i){
a <- data[,i] %>%
  group_by_all() %>%
  summarise(n=n()) %>%
  ungroup() %>%
  mutate(prop=n/sum(n)) %>%
  mutate(type=i)

a[,2:ncol(a)] %>%
  mutate(class=as.character(pull(a[,1]))) %>%
  arrange(desc(class)) %>%
  mutate(lab.ypos = cumsum(prop) - 0.5*prop)

})

ggplot(donut_df, aes(x = 2, y = prop, fill = class)) +
  geom_bar(stat = "identity", color = "white") +
  coord_polar(theta = "y", start = 0)+
  geom_text(aes(y = lab.ypos,
                label =
                  ifelse(prop<0.03,"",paste0(round(prop*100),"%") )
                ),
                color = "white")+
  theme_void()+
  xlim(0.5, 2.5) +
  facet_wrap(~type) +
  scale_fill_manual(values=c("#0f3057", "#00587a", "#008891", "#e7e7de",
                             "#db6400", "#8db596", "#bedbbb", "#ac4b1c",
                             "#87431d", "#83142c", "#aa7070", "#66779c", "#b2deec"))

# Kaplan-Meier estimates of the probability of survival over time
km_fit <- survfit(Surv(surv_time, status) ~ 1, data=data)
autoplot(km_fit) +
  ggtitle("Kaplan-Meier␣estimates␣of␣\n␣the␣probability␣of␣survival␣over␣time") +
  theme_economist()


# we now look at survival curves by gender
# we can see that those without specified gender have the highest probability of
# survival
# followed by females
km_trt_fit <- survfit(Surv(surv_time, status) ~ gender, data=data)
autoplot(km_trt_fit) +
  labs(title = "Kaplan-Meier␣estimates␣of␣the␣probability␣of␣survival␣over␣time",
       subtitle = "Survival␣curves␣by␣gender") +
  theme_economist() +
  scale_fill_economist() +
  scale_color_economist()


# now we look at the differences between different age groups
km_AG_fit <- survfit(Surv(surv_time, status) ~ age_group, data=data)
autoplot(km_AG_fit) +
  labs(title = "Kaplan-Meier␣estimates␣of␣the␣probability␣of␣survival␣over␣time",
       subtitle = "Survival␣curves␣by␣age␣group") +
  theme_economist() +
  scale_fill_economist() +
  scale_color_economist()

# now we look at the differences between different between who has a promotion code
# and who doesn't
km_prom_fit <- survfit(Surv(surv_time, status) ~ prom_code, data=data)
autoplot(km_prom_fit) +
  labs(title = "Kaplan-Meier␣estimates␣of␣the␣probability␣of␣survival␣over␣time",
       subtitle = "Survival␣curves␣by␣promotion␣code␣presence") +
  theme_economist() +
  scale_fill_economist() +
  scale_color_economist()


# now we look at the survival curve between who makes extra attempts and who doesn't
km_att_fit <- survfit(Surv(surv_time, status) ~ extra_attempts_dummy, data=data)
# interesting intersection between the curves
# making extra attempts before the intersection means slightly higher probability
# of survival
# after the intersection slightly lower probability of survival
autoplot(km_att_fit) +
  theme_economist() +
  scale_fill_economist() +
  scale_colour_economist() +
  labs(title = "Kaplan-Meier␣estimates␣of␣the␣probability␣of␣survival␣over␣time",
       subtitle = "Extra␣attempts␣dummy")

# now we compare groups where the main_extra_attempt occurs
# remove the ones with a too high confidence interval and coclude
data_reduced <- data %>%
  filter(!main_extra_attempt %in%
           c("antiriciclaggio","rapporto","contract-subscription","conclude"))

# there are still a lot of overlapps with confidence intervals though
km_grp_fit <- survfit(Surv(surv_time, status) ~ main_extra_attempt, data=data_reduced)
autoplot(km_grp_fit) +
theme_economist() +
  scale_fill_economist() +
  scale_colour_economist() +
  labs(title = "Kaplan-Meier␣estimates␣of␣the␣probability␣of␣survival␣over␣time",
       subtitle = "Where␣the␣main␣extra␣attempt␣occurs")
```

```
#-------- LOGISTIC REGRESSION ---------------

inputData <- data[2:13]

# Create Training Data
input_ones <- inputData[which(inputData$status == 1), ]  # all 1's
input_zeros <- inputData[which(inputData$status == 0), ]  # all 0's
set.seed(100)  # for repeatability of samples

input_ones_training_rows <- sample(1:nrow(input_ones), 0.7*nrow(input_ones))
# 1's for training

input_zeros_training_rows <- sample(1:nrow(input_zeros), 0.7*nrow(input_ones))
# 0's for training. Pick as many 0's as 1's

training_ones <- input_ones[input_ones_training_rows, ]
training_zeros <- input_zeros[input_zeros_training_rows, ]
trainingData <- rbind(training_ones, training_zeros)  # row bind the 1's and 0's

# Create Test Data
test_ones <- input_ones[-input_ones_training_rows, ]
test_zeros <- input_zeros[-input_zeros_training_rows, ]
testData <- rbind(test_ones, test_zeros)  # row bind the 1's and 0's

logitMod <- glm(status ~ stuck + surv_time + extra_attempts +
                main_extra_attempt + mean_time +  gender + prom_code,
                data=trainingData,
                family=binomial(link="logit"))

predicted <- predict(logitMod, testData, type="response")  # predicted scores
predicted

library(InformationValue)
optCutOff <- optimalCutoff(testData$status, predicted)[1]

summary(logitMod)

library(car)
vif(logitMod)

misClassError(testData$status, predicted, threshold = optCutOff)

plotROC(testData$status, predicted)

Concordance(testData$status, predicted)

sensitivity(testData$status, predicted, threshold = optCutOff)

specificity(testData$status, predicted, threshold = optCutOff)
```