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

October 27, 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 he is stuck and try to learn something from this kind of data. We will try to study the percentage of people that will really conclude the process (with respect to those who are stuck in the process) and try to build a model for predict if a people will conclude or not the subscription (basing on his/her behavior). The data were pre-anonymized by the company that provided them.

***K*eywords** R · Statistics · Survival Analysis · Logistic Regression · More

1 Introduction

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

- ID: unique identifier of the event;
- COMPLETED_STEP: identify the [completed] step of the event in JSON format;
- DATE_EVENT: it contains the date and the time of the event;
- NETWORK_ID: represent the network by which the event occurred;
- USER_ID: unique identifier of the user;

Note that we speak about event because each subscription step will trigger an event on the database.
Each steps represent 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 personal data are requested;
- codice fiscale: here the user should provide his/her fiscal code;
- antiriciclaggio: this step refer to the anti-money laundering and is ask to the user if he/she is a politics or an "exposed" person and the source of his/her money;
- conclude: if a user arrives here he/she has concluded the subscription process;
- contract-activation, contract-subsritpion.

Starting from this dataset one can think about which kind of information it can deliver, but for see it we should run some operation over it. As matter of fact we have grouped users with their id in order to calculate all of this information.
In more details, we created a new dataset which include:

- 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 unit;
- status: is a flag that will assume value 1 if a user has arrived to the step "conclude";
- extra_attempts: indicate if the user has done other attempt of subscription;
- main_extra_attempt: it will assume a value different from "none" if the previous item is different from zero and it represent the last step in which he/she is stuck;
- stuck: contain 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 step (note that he can go back, restart, go forward, etc.) he/she does.

All our analysis is based on this new dataset.

2 Survival analysis

In this chapter we will discuss about the survival analysis with the aim to see how many people went all the way through the subscription process (along a time scale) or where they are stuck. Said that, we can also see if a person will conclude the subscription after a while of begin stuck.
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 in Figure 1 where the client are stuck in the subscription process.

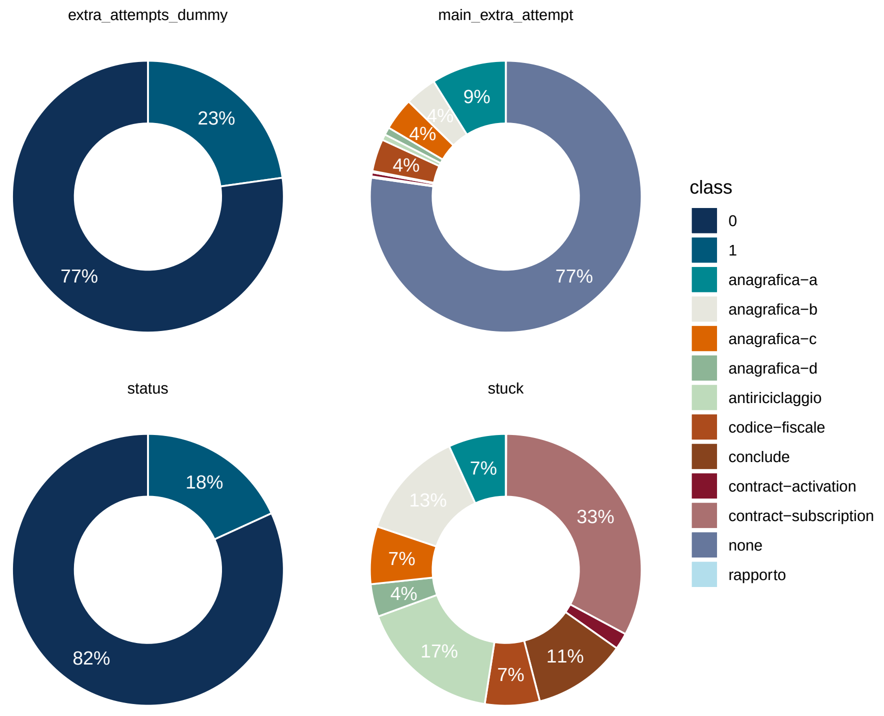

Figure 1: Donut graph of stuck users

Before discuss in detail our survival analysis, we should say few word about which kind of method we used. We used the "Kaplan–Meier estimator" that is a method used (especially in medical research) to estimate the probability that

something will survive over time. Here, by surviving, we means that a user is still in the 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 decrease).

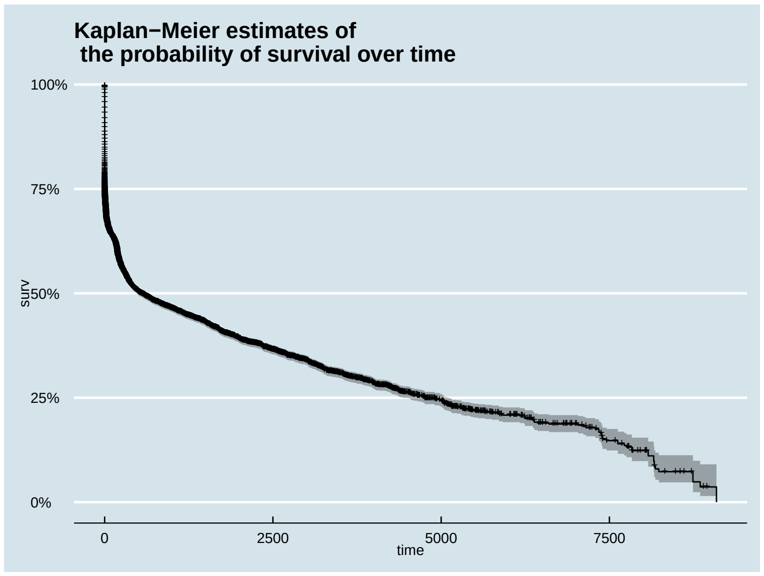

Figure 2: Probability of survival over time

Said that, we can now we look at the survival curve between who makes extra attempts and who does not. As shown in Figure 3, who makes extra attempts is more likely not to conclude (since the survival rate is lower) and there are small overlaps between the curves.

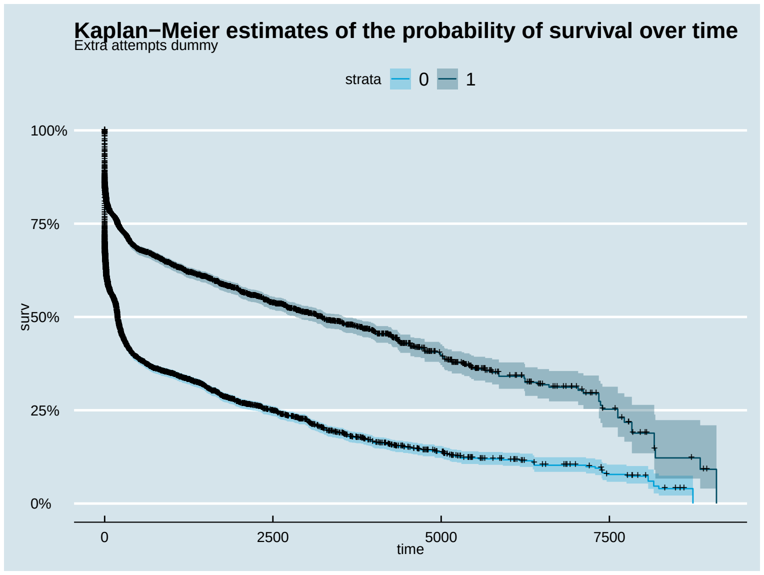

Figure 3: Probability of survival over time with extra attempt

Now, we can compare groups where the main stuck occurs and remove the ones with a too high confidence interval but there are still a lot of overlapps with confidence intervals though, as we can see in Figure 4.

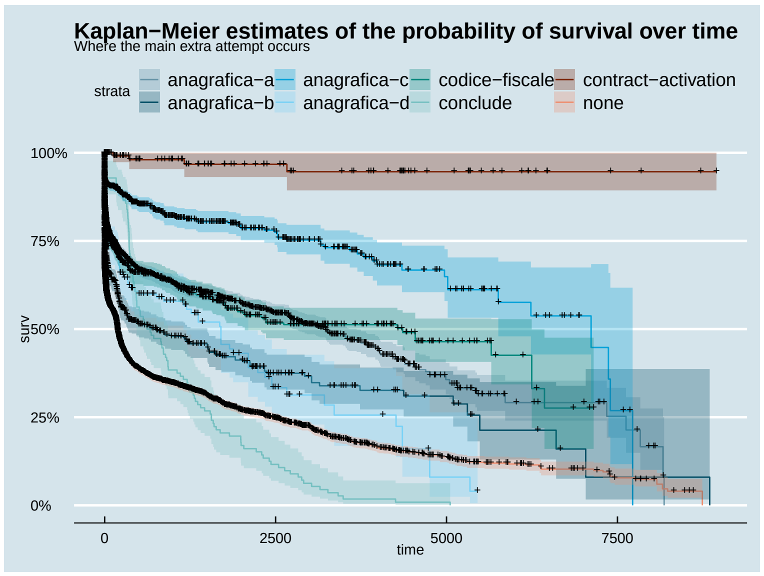

Figure 4: Total probability of survival over time with extra attempt

If we look at Figure 4, we can see that who does not perform any extra attempt has a lower probability of survive 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 around 7500 we can see that there is some overlappings. Finally, we look at where the client was last stuck in Figure 5.

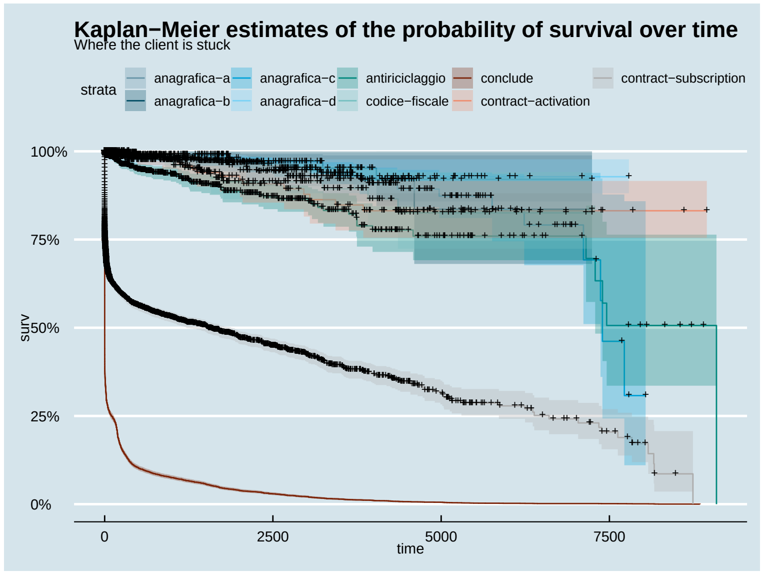

Figure 5: Probability of survival over time where the client is stuck

3 Predict the probability of conclude the subscription

Furthermore, we can expand our analysis by creating a model for predicting the probability that a user will conclude the subscription process basing on his/her behavior.

Firstly we choose to use the Logistic Regression because it is particular indicated in the categorical case, i.e. he will or will not conclude the process. We start by dividing the dataset in train and test part with the rule of 7-3, moreover with divide this data by status (that can assume only value 0 or 1).
After having created this to sub-dataset (i.e. the train and the test set), we can define our model with "status" as response variable and "stuck", "surv_time", "extra_attempts", "main_extra_attempt" and "mean_time" as explanatory variable. 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, family = binomial(link = "logit"), data = trainingData)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.60610  -0.27766  -0.05337   0.00008   3.13124

Coefficients:
                                  Estimate Std. Error z value Pr(>|z|)
(Intercept)                       5.061e+00  5.128e-01  -9.869  < 2e-16 ***
stuckanagrafica-b                 1.910e-02  6.145e-01   0.031 0.975207
stuckanagrafica-c                 1.710e+00  5.436e-01   3.145 0.001660 **
stuckanagrafica-d                 1.735e+00  5.715e-01   3.035 0.002403 **
stuckantiriciclaggio              1.928e+00  5.165e-01   3.733 0.000189 ***
stuckcodice-fiscale               6.213e-01  6.176e-01   1.006 0.314445
stuckconclude                     2.464e+01  1.531e+02   0.161 0.872201
stuckcontract-activation          1.890e+00  6.389e-01   2.958 0.003100 **
stuckcontract-subscription        5.175e+00  5.025e-01  10.297  < 2e-16 ***
stuckrapporto                     3.147e+00  1.196e+00   2.631 0.008504 **
surv_time                         1.104e-04  6.145e-05   1.796 0.072465 .
extra_attempts                    2.448e-02  2.158e-02   1.134 0.256665
main_extra_attemptanagrafica-b   -8.108e-02  1.519e-01  -0.534 0.593389
main_extra_attemptanagrafica-c    1.833e-01  1.829e-01   1.002 0.316342
main_extra_attemptanagrafica-d   -1.393e-01  2.677e-01  -0.521 0.602669
main_extra_attemptantiricicl.     4.593e-01  3.008e-01   1.527 0.126726
main_extra_attemptcodice-fisc.    2.672e-01  1.798e-01   1.487 0.137122
main_extra_attemptconclude       -2.390e-01  1.256e+03   0.000 0.999848
main_extra_attemptcontract-act.  -6.576e-02  7.492e-01  -0.088 0.930052
main_extra_attemptcontract-sub.   3.223e+00  1.022e+00   3.153 0.001615 **
main_extra_attemptnone           -1.040e-01  1.198e-01  -0.868 0.385583
main_extra_attemptrapporto        1.986e+01  4.746e+03   0.004 0.996662
mean_time                         9.300e-04  3.009e-04   3.091 0.001997 **
---
Signif. codes:  0 "***" 0.001 "**" 0.01 "*" 0.05 "." 0.1 " " 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 22322.1  on 16101  degrees of freedom
Residual deviance:  8919.8  on 16079  degrees of freedom
AIC: 8965.8

Number of Fisher Scoring iterations: 18
```

As we can see, not all the variable are really explanatory for our estimate.
We can then compute the optimal score that minimizes the misclassification error for the above model, 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	1.713298	9	1.030364
surv_time	2.900087	1	1.702964
extra_attempts	2.125932	1	1.458057
main_extra_attempt	3.374714	10	1.062703
mean_time	2.884597	1	1.698410

Then, we can analyze the misclassification error, that is the percentage mismatch of predcited vs actuals, irrespective of 1's or 0's. The lower the misclassification error, the better is your model. Given our optimal cutoff value, we obtain a misclassification error of 0.0287.
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. So for a good model, the curve should rise steeply, indicating that the TPR (Y-Axis) increases faster than the FPR (X-Axis) as the cutoff score decreases. Greater the area under the ROC curve, better the predictive ability of the model. We can see this plot in Figure 6.

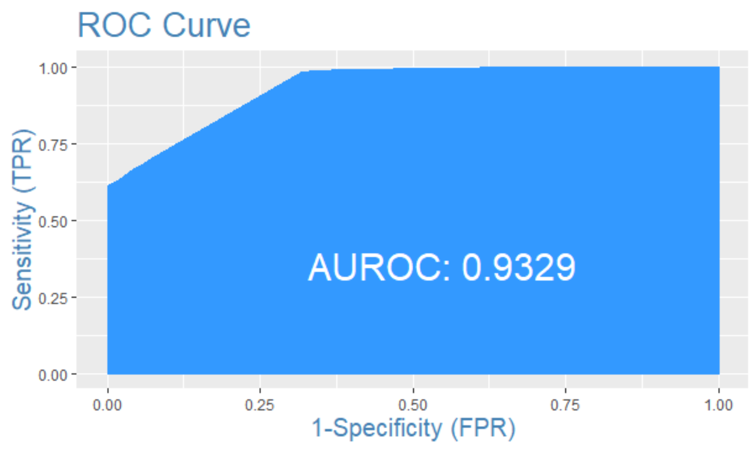

Figure 6: Receiver Operating Characteristics Curve

4 Conclusion

We can analyze the concordance of our estimation which means that the model-calculated-probability-scores of all actual Positive's, (aka Ones) should be greater than the model-calculated-probability-scores of ALL the Negatives (aka 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: 0.9314313, i.e 93%;
- Discordance: 0.0685687, i.e. around 7%;
- 150912230 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 0.608809, i.e. 60%;
- Specificity of 0.9999085, i.e. around 100%;

Now, one can ask himself why this model gives us so high precision and why the survival analysis does not give us so strange result. The answer to that is really simple: lack of data. Keep in mind that the omitted data are sensitive ones, so if we really want to deal with this kind of data we should pay attention to the Regulation (EU) 2016/679 (General Data Protection Regulation).
So, basically, we have successfully analyze the survival rate of a customer willing to subscribe to the system along with the creation a model that can help us to know the probability that a certain person will or will not conclude the process, without using any sensitive information about the users.

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()

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

# some users just have one conclude without the previous steps

# 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)

  time_diff <- difftime(max(x$DATE_EVENT), min(x$DATE_EVENT),
           units = "hours")

  mean_time <- time_diff/nrow(x)

  extra_attempts <- x %>%
    group_by(COMPLETED_STEP) %>%
    filter(attempt == max(attempt)) %>%
    mutate(check = ifelse(attempt==1,0,attempt-1)) %>%
    ungroup() %>%
    summarise(extra_attempt = sum(check)) %>% pull()

  # 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_STE

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

  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
        )
      )

  # 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))

# many outliers for survival time
# few clients go through the system for a long time
boxplot(data$surv_time,main="Many␣outliers␣going␣throuh␣the␣\n␣system␣for␣a␣long␣time")

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

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)

summary(km_fit)

# as time passes the probability of concluding increases
autoplot(km_fit) +
  ggtitle("Kaplan-Meier␣estimates␣of␣\n␣the␣probability␣of␣survival␣over␣time") +
  theme_economist()

# now we look at the survival curve between who makes extra attempts and who doesn't
km_grp_fit <- survfit(Surv(surv_time, status) ~ extra_attempts_dummy, data=data)
# who makes extra attempts is more likely not to conclude and there are
# small overlaps between the curves
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 = "Extra␣attempts␣dummy")

summary(km_grp_fit)

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

# 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")

summary(km_grp_fit)

# Look at where the client was last stuck
data_reduced <- data %>%
  filter(!stuck %in%
           c("rapporto"))
```

```
km_grp_fit <- survfit(Surv(surv_time, status) ~ stuck, 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␣client␣is␣stuck") +
  theme(legend.text = element_text(size=10))

summary(km_grp_fit)


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

inputData <- data[2:7]

# 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

library(smbinning)

logitMod <- glm(status ~ stuck + surv_time + extra_attempts + main_extra_attempt + mean_tin

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)
```