Appendix

B

R SCRIPT

The statistical analysis of this study used the R language. To facilitate future analyses and to better understand the analyses performed in this study, we attach pseudocode of the commands used to analyze the dimensions and features from the perspective of popularity and positive evaluations.

We divide the script into the analysis phases, bringing the libraries used and their respective commands.

1. Read file

```
library(readr)
read.csv(file = file.csv)
```

2. Definition of the independent feature (Popularity)

```
Name–Table \$ Name–Collumn \textless−
as.factor(Name–Table \$ independent−feature)
```

3. Creating the Logistic Regression of the dimension(s)

```
Regression−Name
\textless−glm(independent−feature
~log(1 + dimension−01)+
...
log(1+dimension−N)+
data = Table,
family = binomial(link = "logit"))
```

4. Calculating AUC and applying bootstrapping technique on the dimensions

```
library(ROCR)
performance(prediction

(Regression—Name$linear.
predictors, Regression—Name$data
$independent—feature),
measure = ‘‘auc")@y.values[[1]]

library(dplyr)
auc—list=Null
for (i in 1:100) {
    amostra \textless— sample_n(Table,81730,replace = TRUE)
    boot \textless— glm(independent—feature ~

    ~log(1 + dimension—01)+
    ...
    log(1+dimension—N)+
    ,data = amostra,
    family = binomial(link = "logit"),
    x = TRUE, y = TRUE)

    auc—list[i] \textless— performance(prediction(boot
    $linear.predictors,
    boot$data$independent—feature),

    measure = "auc")@y.values[[1]]
}
mean(auc—list)
```

5. Create an AUC rank

```
library(ScottKnottESD)
sk \textless— sk_esd(cbind(auc—lists))
plot(sk, ylim = c(0.7,1), ylab=‘‘AUC", xlab=‘‘Dimensions",
main=‘‘Scott—Knott effect size difference test — Popularity")
```

6. Verify the correlation between the features

```
library(Hmisc)
cor—features \textless— varclus(as.matrix(num—matrix),
c=‘‘spearman")
write.table(correlacao_features_qual[[‘‘sim"]],
file = ‘‘file.csv", sep = ‘‘t", na = ‘‘", quote = FALSE)
```

7. Reducing non-significant features

```
library(Hmisc)
redun(~features\$feature-01+
    ...
    feature-N+,
    data=features, subset=NULL, r2 = 0.9,
    type = c("ordinary", "adjusted"),
    nk = 0, tlinear = FALSE,
    allcat=FALSE, minfreq=0,
    iterms=FALSE, pc=FALSE, pr = FALSE)
```

8. Creating the Logistic Regression of the Features(s)

```
log-redun-feature <- rms::Glm(independent-feature~
                                  log(1+feature-01)+
                                  ... +
                                  log(1+feature-N)+
                       data = Table,
                       family = binomial(link = ``logit"),
                       x = TRUE, y = TRUE)
```

9. Estimate regression coefficient

```
set.seed(1)
library(rms)
bootcov(log_redun_feature, B=100)
```

10. Explanatory power of features

```
anova(log-redun-feature, type=c("II","III", 2, 3),
test.statistic=c("Wald"))
```

11. Explanatory power of the dimensions

```
library('lmtest')
log-without-dimension-01 \textless- glm(independent-feature~
                             log(1+feature-01)+
                            ... +
                            log(1+feature-N)+
                            data = Table,
                           family = binomial(link = ``logit"),
                           x = TRUE, y = TRUE)

lrtest(log-without-dimension-01,log-redun-feature)
The previous steps are performed for each dimension.
```

12. Backward Resource Selection

```
library(rms)
fastbw(log-redun-feature, rule=c(``aic", ``p"),
   type=c(``residual", ``individual", ``total"),
   sls=.05, aics=0, eps=1e-9,
   k.aic=2, force=NULL)
```

13. Create Final Logistic Model

```
final-model <- rms::Glm(independent-feature~
                              log(1+feature-01)+
                              ... +
                              log(1+feature-N)+
                        data = Table,
                        family = binomial(link = ``logit"),
                        x = TRUE, y = TRUE)
```

14. Estimate regression coefficient

```
set.seed(1)
library(rms)
bootcov(final-model, B=100)
```

15. Create nomogram

```
library(rms)
ddist <- datadist(Table);
options(datadist='ddist')
nom <- nomogram(final-model,
          fun = function(x){1/(1+exp(-x))},
          fun.at = c(.1,.5,.9,1),
          conf.int = c(0.1,0.7),
          funlabel = ``Name-Label",
          nint = 10
          )

plot(nom, xfrac=.45, cex.axis = 1.1)
```