rappi_challenge

Emiliano Ramírez

2022-07-14

Leemos base de datos y proporcionamos formato adecuado ya que tenemos columnas JSON y otros contratiempos con la base. También, procesamos variables

```
df <- read.csv("raw/ds_challenge_data.csv")


#vemos que existe una columna en formato json por lo que la formatearemos

#creamos función para deparse de collumna en formato json
ParseJSONColumn <- function(x)  {
  str_c("[ ", str_c(x, collapse = ",", sep=" "), " ]")  %>%
    fromJSON(flatten = T) %>%
    as_tibble()
}

#seleccionamos columna que tiene formato json
dispositivo <- df %>% select(dispositivo)

dispositivo_unj <- dispositivo  %>%
  map_dfc(.f = ParseJSONColumn)


#eliminamos la variable model porque es constante en toda la base
df <- bind_cols(df, dispositivo_unj) %>% select(-c(dispositivo, model))

#notemos que tenemos varias variables que deben ser declaradas como categóricas
#para su correcto tratamiento.

#Estas variables son: genero, estableciemiento, ciudad, tipo tarjeta (tipo_tc),
#status_txn, is_prime, fraude, device_Score y os.

#damos un poco de limipieza a algunas variables antes de su procesamiento
df <- df %>% mutate(genero=ifelse(genero=="--", NA, genero),
                    establecimiento=ifelse(establecimiento=="N/A",NA,
                                           ifelse(establecimiento=="",NA,establecimiento)),
                    ciudad=ifelse(ciudad=="", NA, ifelse(ciudad=="N/A",NA,ciudad)),
                    os=ifelse(os==".", NA, os))

# en genero: 1 mujer 0 hombre
# factores: establecimiento, ciudad,
# 1 tarjeta fisica  0 virtual
df <- df %>% mutate(d_genero=ifelse(genero=='F',1,ifelse(genero=='M',0,NA)),
```

```
                    establecimiento=as.factor(establecimiento),
                    ciudad=as.factor(ciudad),
                    d_tarj=ifelse(tipo_tc=="Física", 1, 0),
                    status_txn=as.factor(status_txn),
                    d_is_prime=ifelse(is_prime=="True", 1, 0),
                    d_fraude=ifelse(fraude=="True",1,0),
                    os=as.factor(os),
                    fecha=as.Date(dmy(fecha)))
```

Obtenemos un panorama general de la base de datos y sus variables, así como estadísticos descriptivos que nos pueden ayudar a tener un conocimiento inicial de los datos.

```
df %>% skim()
```

Table 1: Data summary

Name	Piped data
Number of rows	26975
Number of columns	21
Column type frequency:	
character	4
Date	1
factor	4
numeric	12
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
genero	2730	0.9	1	1	0	2	0
tipo_tc	0	1.0	6	7	0	2	0
is_prime	0	1.0	4	5	0	2	0
fraude	0	1.0	4	5	0	2	0

Variable type: Date

skim_variable	n_missing	complete_rate	min	max	median	n_unique
fecha	0	1	2020-01-02	2020-01-30	2020-01-16	29

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
establecimiento	10119	0.62	FALSE	5	Res: 3454, Aba: 3415, Sup: 3402, MPa: 3343

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
ciudad	11678	0.57	FALSE	4	Tol: 3997, Gua: 3833, Mer: 3761, Mon: 3706
status_txn	0	1.00	FALSE	3	Ace: 18844, En : 5341, Rec: 2790
os	6715	0.75	FALSE	3	%%: 6808, WEB: 6766, AND: 6686

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
ID_USER	0	1.0	2003.77	1144.63	0.00	1041.00	2006.00	2973.50	3999.00	
monto	0	1.0	499.07	289.31	0.02	246.52	500.50	749.60	999.92	
hora	0	1.0	11.99	6.64	1.00	6.00	12.00	18.00	23.00	
linea_tc	0	1.0	62476.81	21886.89	25000.00	44000.00	62000.00	82000.00	99000.00	
interes_tc	0	1.0	48.22	9.59	32.00	40.00	48.00	57.00	64.00	
dcto	0	1.0	17.47	34.33	0.00	0.00	0.00	18.77	199.36	
cashback	0	1.0	6.26	4.46	0.00	2.79	5.64	8.53	19.99	
device_score	0	1.0	3.00	1.42	1.00	2.00	3.00	4.00	5.00	
d_genero	2730	0.9	0.44	0.50	0.00	0.00	0.00	1.00	1.00	
d_tarj	0	1.0	0.70	0.46	0.00	0.00	1.00	1.00	1.00	
d_is_prime	0	1.0	0.13	0.34	0.00	0.00	0.00	0.00	1.00	
d_fraude	0	1.0	0.03	0.17	0.00	0.00	0.00	0.00	1.00	

```
summary(df$d_genero)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.0000  0.0000  0.0000  0.4424  1.0000  1.0000    2730
```

El 44.2 % de la muestra es del sexo femenino y existen 2730 missings.

```
stat.desc(df$linea_tc)
```

```
##             nbr.val            nbr.null              nbr.na                 min
##       26975.0000000           0.0000000           0.0000000       25000.0000000
##                 max               range                 sum              median
##       99000.0000000       74000.0000000 1685312000.0000000       62000.0000000
##                mean             SE.mean        CI.mean.0.95                 var
##       62476.8118628         133.2610976         261.1986723 479036080.8860460
##             std.dev            coef.var
##       21886.8929016           0.3503203
```

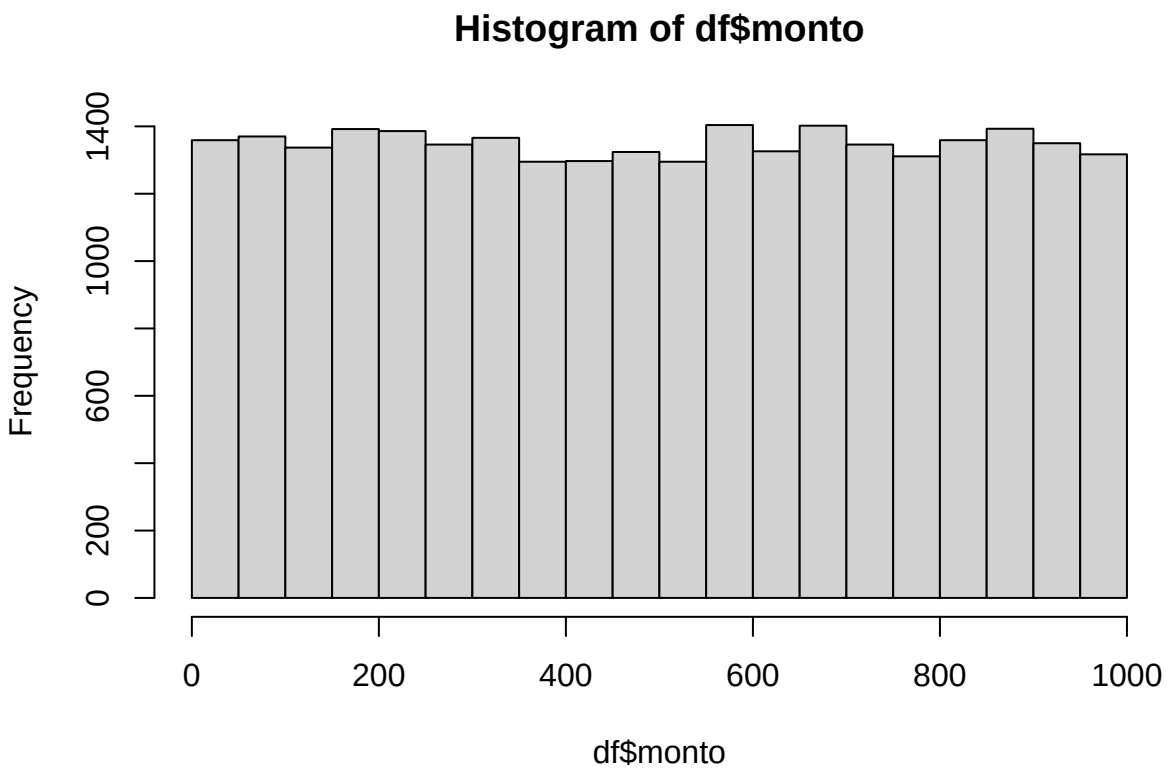
La mediana es casi igual a la media de la distribución de la var de linea de crédito por lo que es factible suponer que la distribución es simétrica, la proporción de la desviación estándar con respecto a la media es del 35%, por lo que las colas no se separan mucho del centro y, por ende, es una muestra homogénea.

```
stat.desc(df$monto)
```

```
##             nbr.val            nbr.null              nbr.na                 min
##     26975.00000000         0.00000000         0.00000000         0.01730251
##                 max               range                 sum              median
```

```
##      999.91776360      999.90046109 13462399.56280527      500.50102160
##              mean           SE.mean      CI.mean.0.95               var
##      499.06949260       1.76149735       3.45262629    83699.99698197
##           std.dev          coef.var
##      289.30951761       0.57969786
```

```
hist(x=df$monto)
```



El monto de las transacciones no supera los 1,000 pesos. Además, su distribución exhibe un comportamiento uniforme y simétrico ya que la media es casi igual que la mediana y su plot lo muestra.

```
#creamos función para sacar la moda
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

df_mode <- df %>% summarise_each(funs = Mode, ID_USER)
```

El usuario con más transacciones es el 1958.

```
df_1958 <- df %>% filter(ID_USER==1958) %>% nrow
```

62 transacciones hizo el usuario 1958, en el periodo de un mes.

Suponemos que el sistema operativo "%%" es diferente a Android pero no es web (como ios o harmonyOS).

```
summary(df$os)
```

```
##      %% ANDROID     WEB    NA's
##    6808    6686    6766    6715
```

La frecuencia de los distintos sistemas operativos está balanceada.

```
summary(df$d_tarj)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  1.0000  0.7008  1.0000  1.0000
```

El 70 por ciento de las compras hechas en esta base son con tarjeta física.

```
summary(df$d_status_txn)
```

```
## Length  Class   Mode
##      0   NULL   NULL
```

Cerca del 70 por ciento de las compras son aceptadas.

```
summary(df$d_is_prime)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.1319  0.0000  1.0000
```

Solo el 13 por ciento de la muestra tiene suscripción prime.

```
summary(df$d_fraude)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.00000 0.00000 0.03003 0.00000 1.00000
```

El 3 por ciento de la muestra está clasificada como compra fraudulenta.

```
summary(df$d_fraude==1 & df$d_status_txn==1)
```

```
##    Mode
## logical
```

El 2 por ciento de las transacciones donde la clasificación era fraudulenta la compra fue aceptada.

```
xtabs(~  genero + fraude, data = df)
```

```
##       fraude
## genero False  True
##      F 10392   334
##      M 13131   388
```

```
xtabs(~  genero + status_txn, data = df)
```

```
##        status_txn
## genero Aceptada En proceso Rechazada
##      F     7462       2128      1136
##      M     9459       2661      1399
```

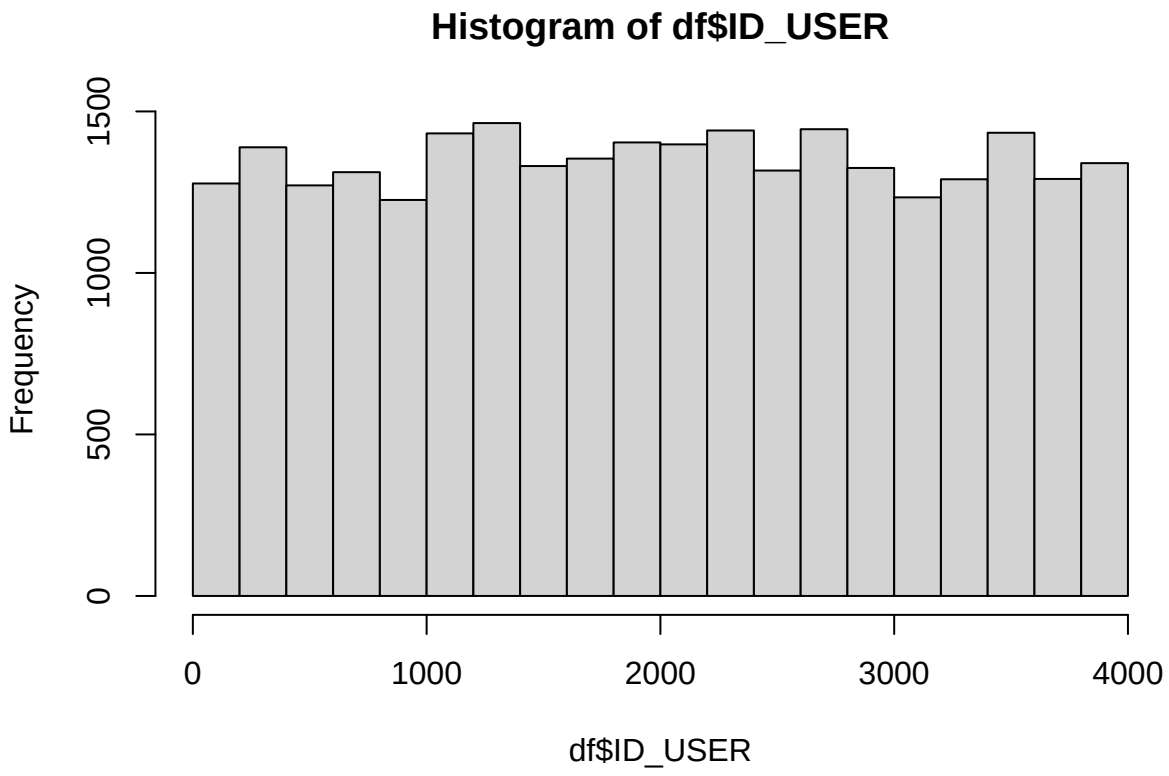
```
xtabs(~  fraude + os, data = df)
```

```
##         os
## fraude    %% ANDROID  WEB
##   False 6595    6470 6577
##   True   213     216  189
```

```
xtabs(~  is_prime + os, data = df)
```

```
##          os
## is_prime   %% ANDROID  WEB
##    False 5937    5813 5865
##    True   871     873  901
```

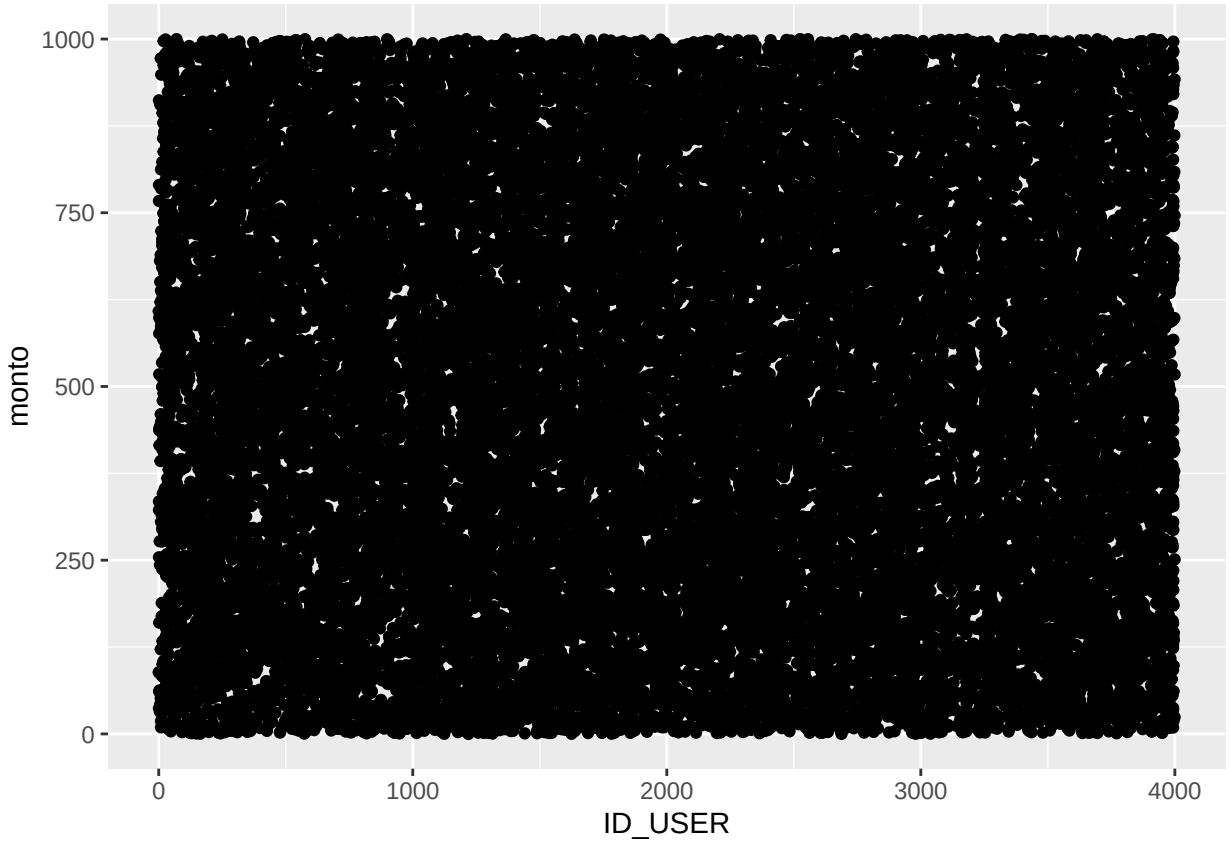
```
hist(x=df$ID_USER)
```



- El sexo masculino tiene 16% más observaciones registradas como fraudulentas.
- El sexo masculino tiene 26% más transacciones aceptadas que el sexo femenino.
- El parecer la frecuencia de las transacciones categorizadas como fraudulentas están balanceadas.
- No parece haber mayor preferencia por la suscripción a través de los distintos os.
- La distribución de la frecuencia de aparición de los usuarios muestra un comportamiento uniforme.

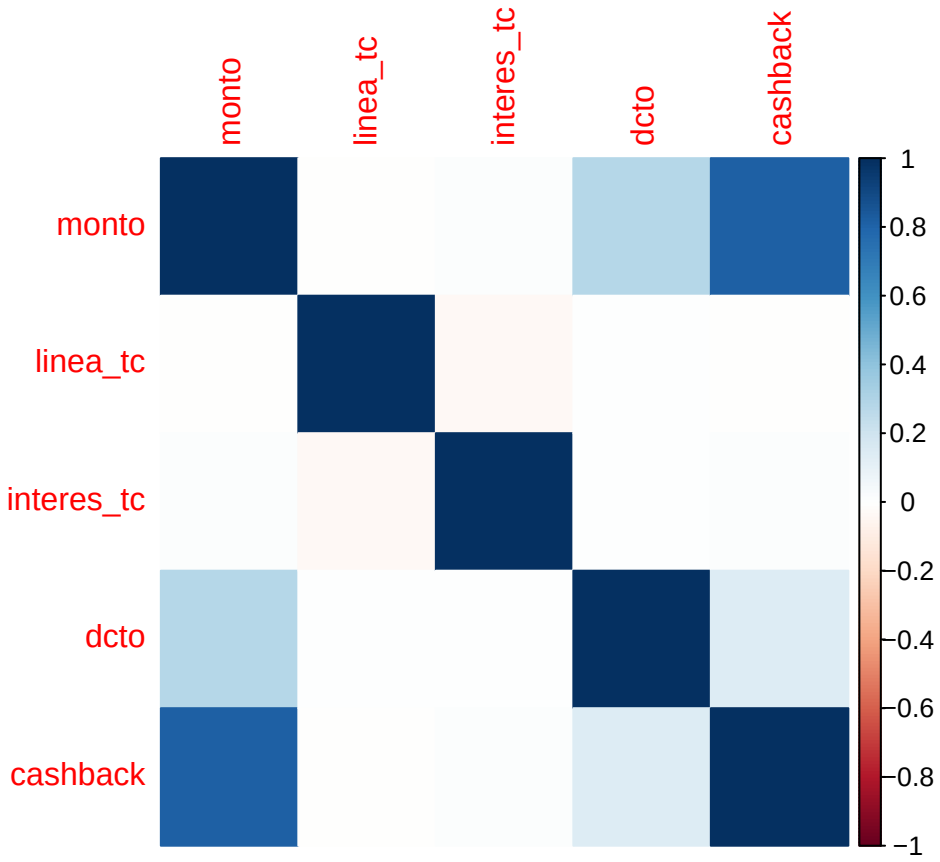
¿Existe alguna concentración de monto de la transacción de algún usuario?

```
ggplot(df, aes(x=ID_USER, y=monto)) +
  geom_point() +
  theme(legend.position="none")
```



Ahora creemos una matriz de correlación lineal para ver si exsite estructura de dependencia linealentre las variables numéricas.

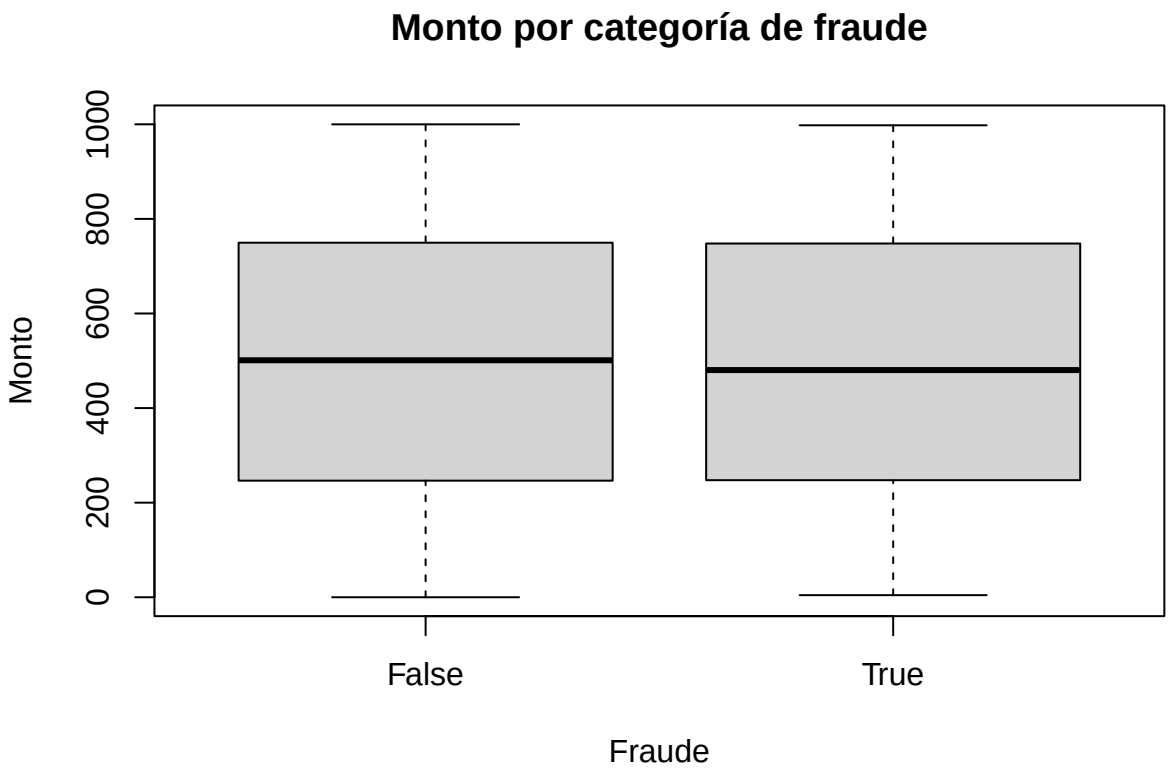
```
df_correlations <- df %>% select(c(monto, linea_tc, interes_tc, dcto,
                                   cashback))

matrix_df <- as.matrix(df_correlations)
mat_corr <- cor(matrix_df)
corrplot(mat_corr, method="color")
```



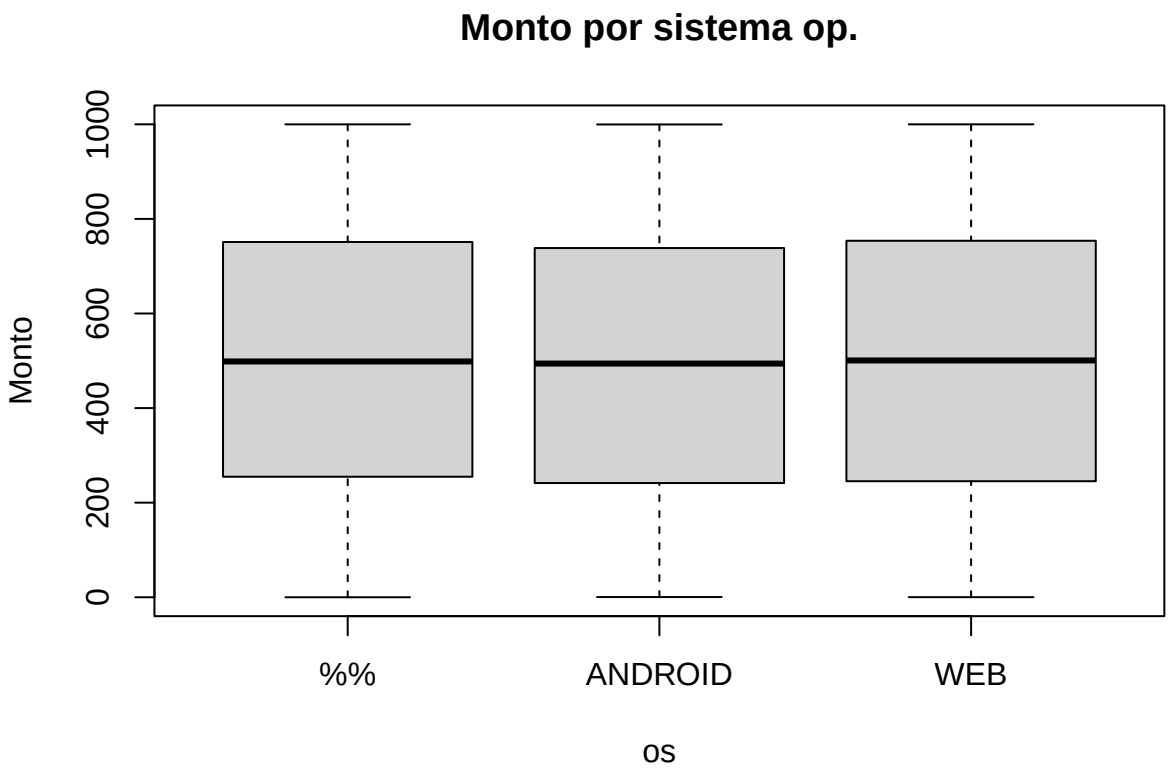
Observando la matriz de correlaciones no existe correlación lineal evidente entre las variables numéricas.

Para observar el 'efecto' de las variables categóricas en las variables numéricas haremos los siguientes boxplots.

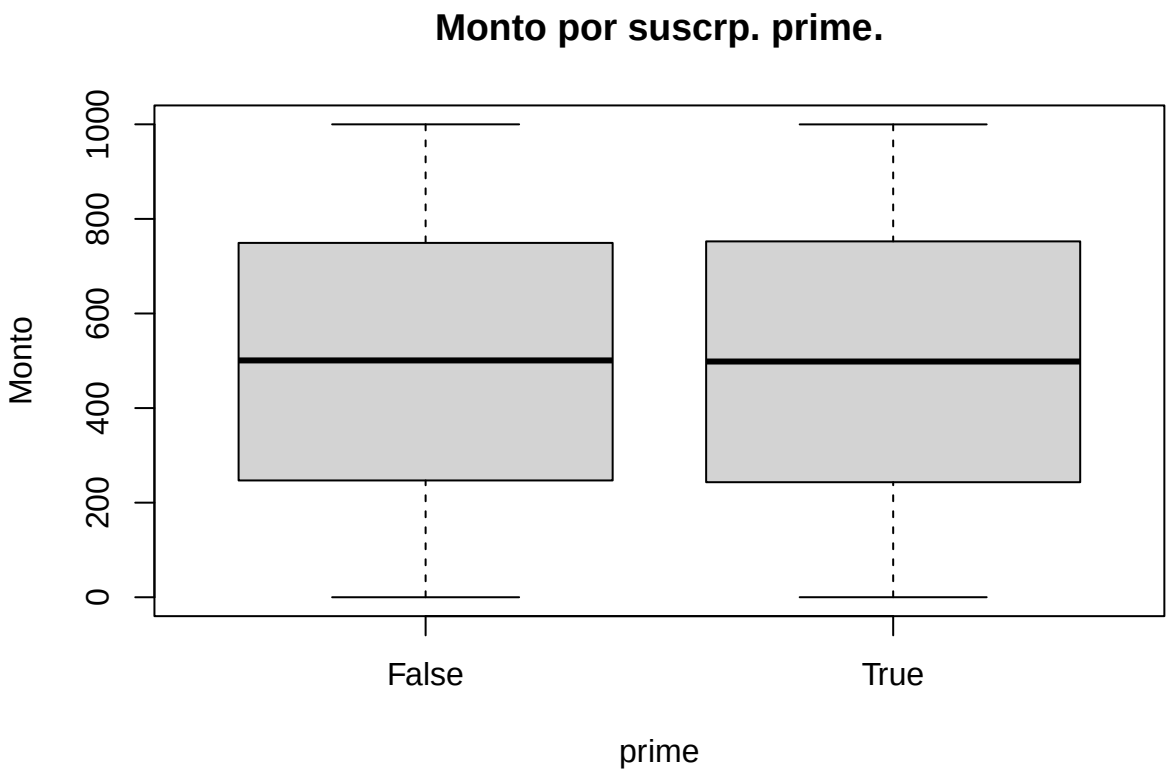
```
boxplot(df$monto ~ df$fraude, main='Monto por categoría de fraude',xlab='Fraude',ylab='Monto')
```



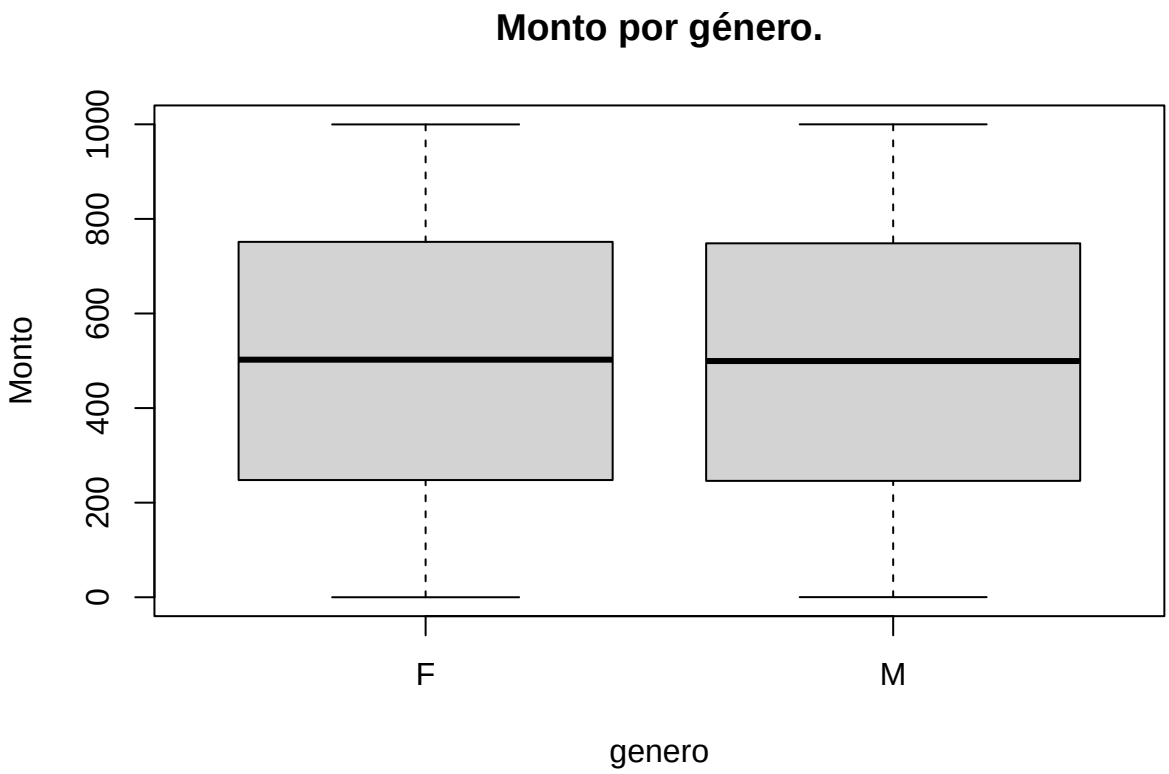

```
boxplot(df$monto ~ df$os, main='Monto por sistema op.',xlab='os',ylab='Monto')
```



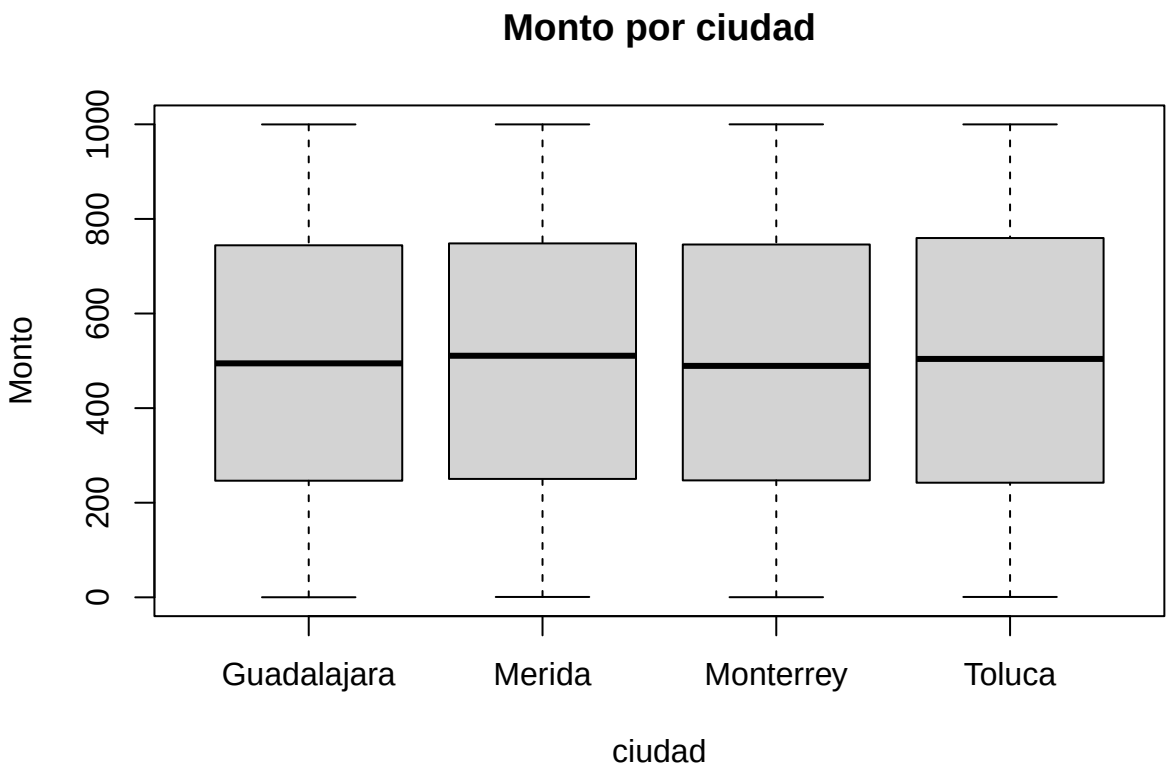

```
boxplot(df$monto ~ df$is_prime, main='Monto por suscrp. prime.',xlab='prime',ylab='Monto')
```



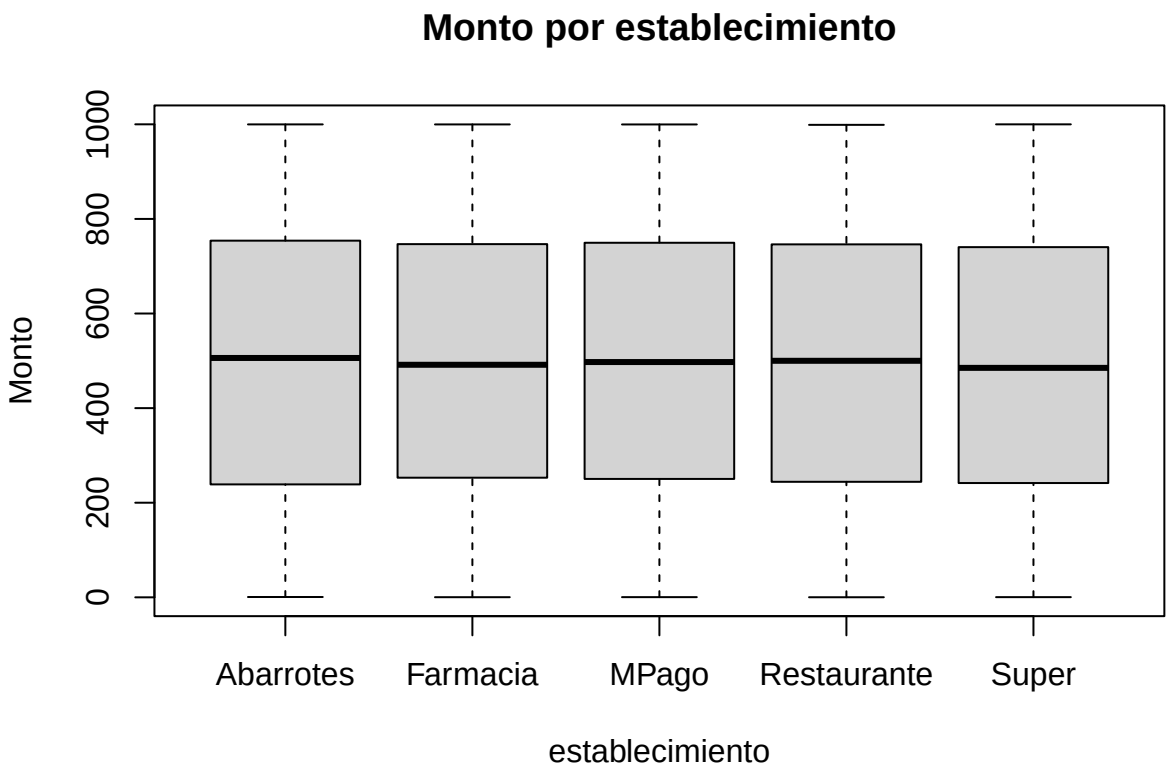

```
boxplot(df$monto ~ df$genero, main='Monto por género.',xlab='genero',ylab='Monto')
```



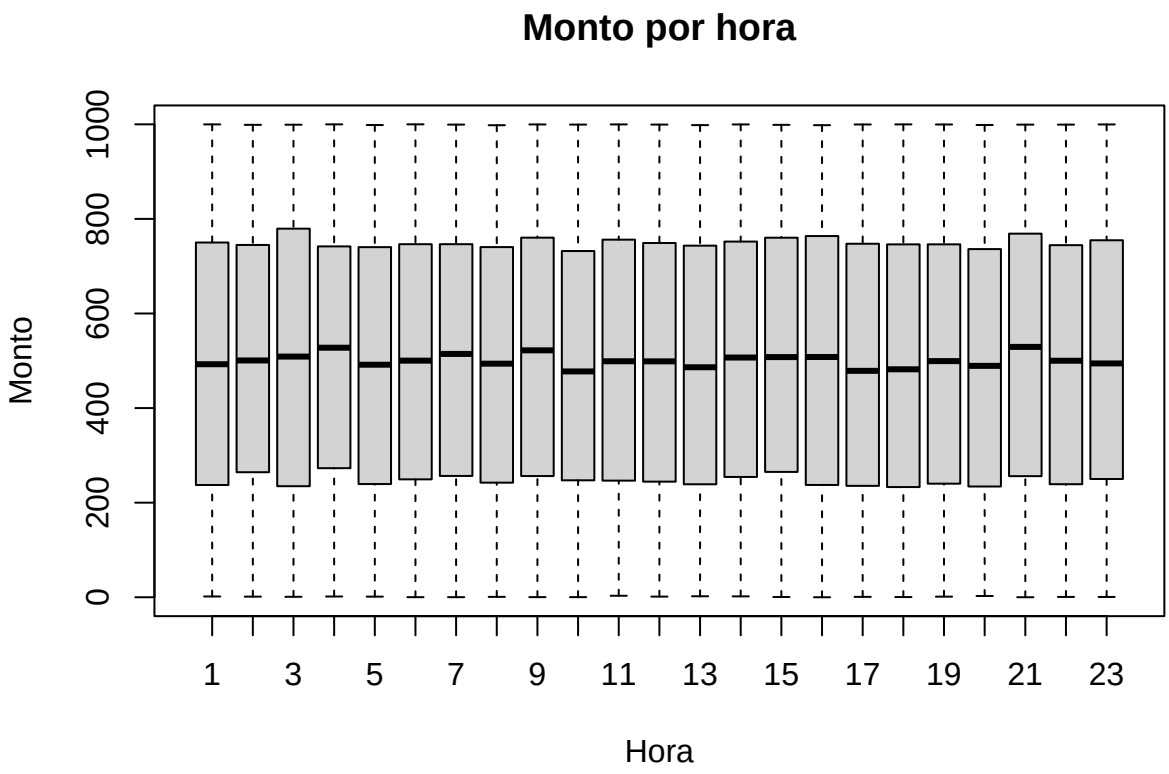

```
boxplot(df$monto ~ df$ciudad, main='Monto por ciudad',xlab='ciudad',ylab='Monto')
```



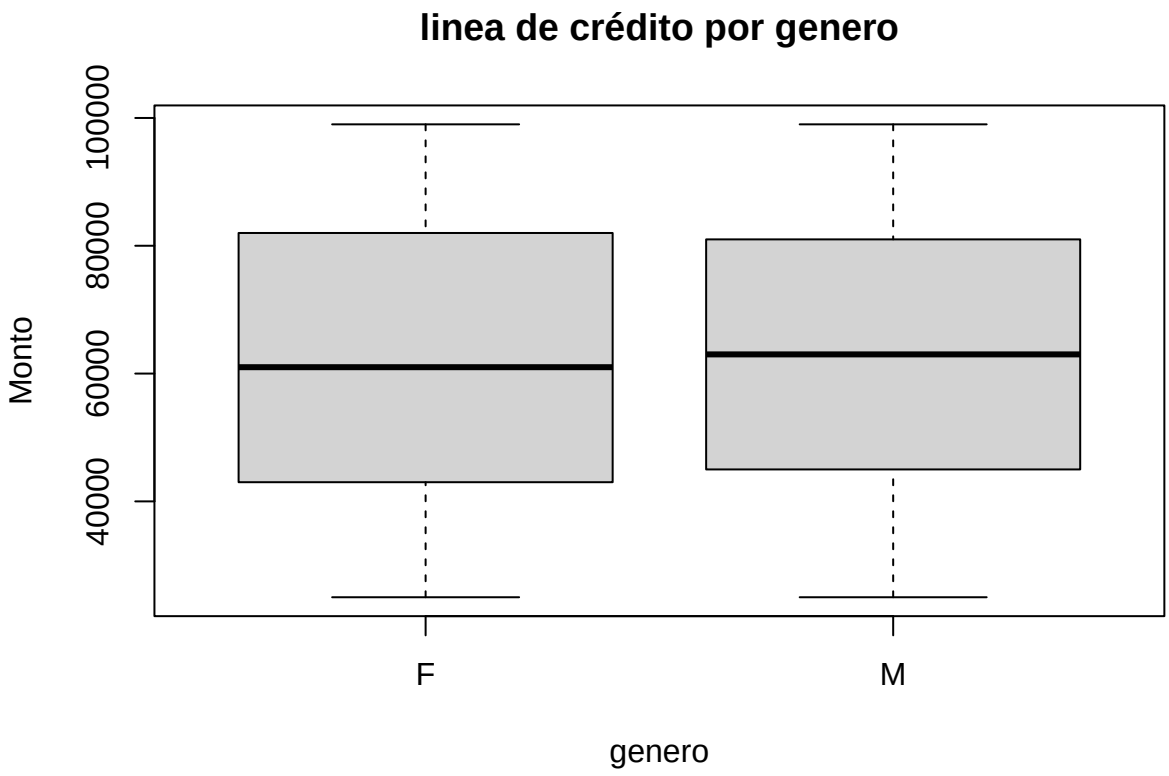

```
boxplot(df$monto ~ df$establecimiento, main='Monto por establecimiento',xlab='establecimiento',ylab='Mo
```



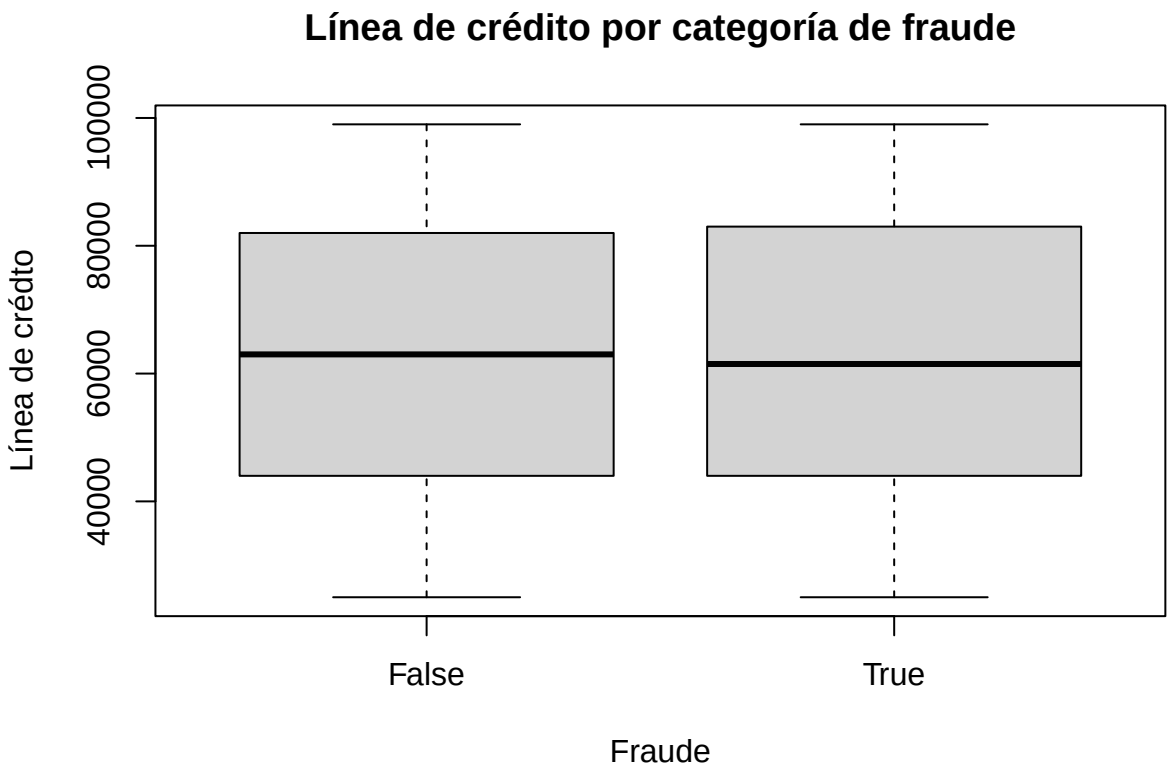

```
boxplot(df$monto ~ df$hora, main='Monto por hora',xlab='Hora',ylab='Monto')
```



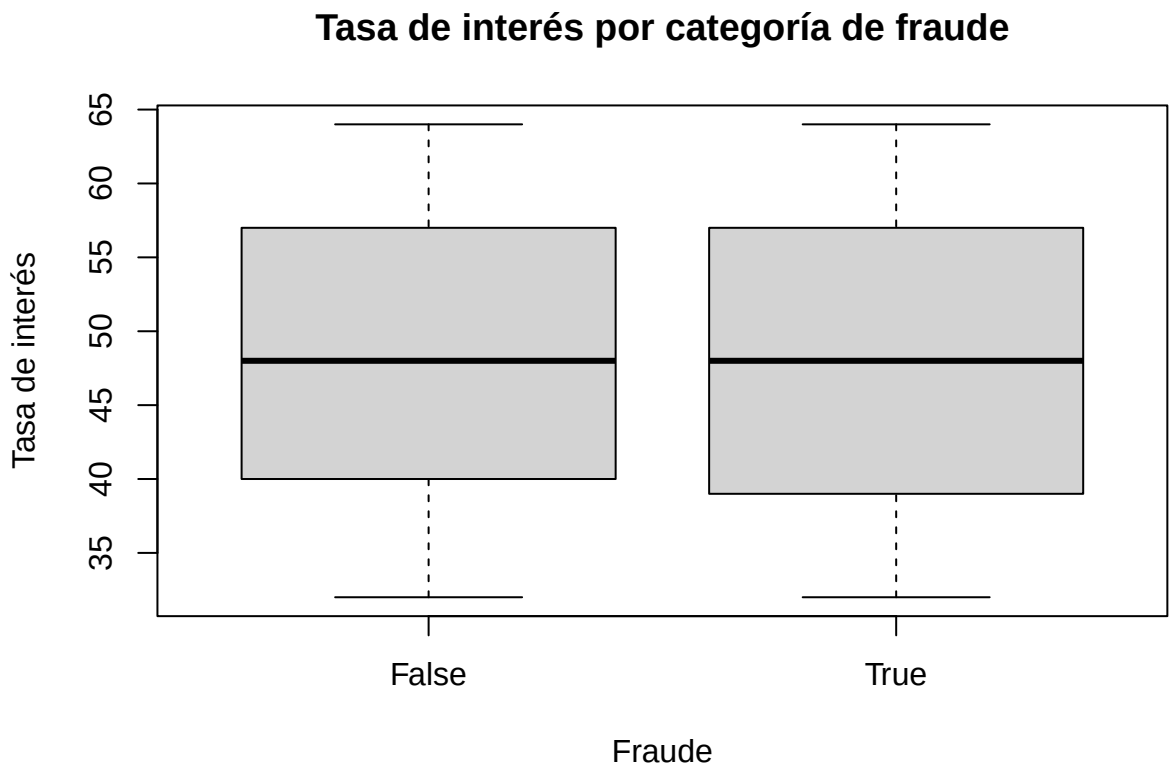

```
boxplot(df$linea_tc ~ df$genero, main='linea de crédito por genero',xlab='genero',ylab='Monto')
```




```
boxplot(df$linea_tc ~ df$fraude, main='Línea de crédito por categoría de fraude',xlab='Fraude',ylab='Lí
```




```
boxplot(df$interes_tc ~ df$fraude, main='Tasa de interés por categoría de fraude',xlab='Fraude',ylab='Ta
```



Al parecer todos los grupos que se generan están balanceados en la muestra.

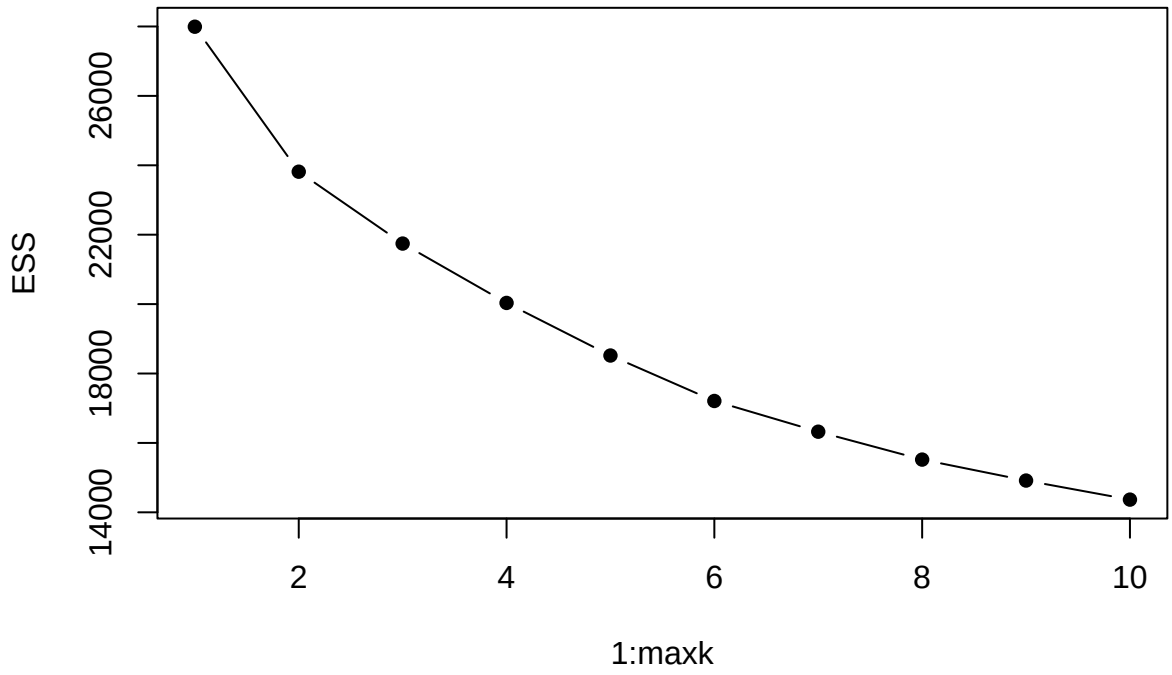
En conclusión, no se encontraron patrones o indicios de algún mecanismo extraño que esté sucediendo en la base.

Procedemos a hacer la clasificación de los usuarios. El algoritmo que se usará es un algoritmo de conglomerados no jerárquico de k-medias (para variables) numéricas.

```
df_num <- df %>% select(ID_USER, monto, hora, linea_tc, interes_tc, dcto, cashback, device_score)


# agreagamos los datos a nivel usuario en la base de vars num
df_prom <- df_num %>%
  group_by(ID_USER) %>%
  summarise_all(mean) %>%
  as.data.frame() %>% select(-ID_USER)

row.names(df_prom) <- df_prom$ID_USER


#ahora estandaricemos las variables ya que tienen distintas escalas entre sí
df_prom <- scale(df_prom)
```

Hacemos pruebas para determinar cuántos clusters hacer.

```r
#gráfica de codo
maxk <- 10 #número de clusters a considerar máximo
ESS <- sapply(1:maxk,function(k)kmeans(df_prom,k,nstart=10)$tot.withinss)
plot(1:maxk,ESS, type="b",pch=16)
```



La gráfica de codo nos arroja un dibujo donde podemos considerar 2 clusters.

Ahora, veamos que nos arroja la función NbClust. Esta función considera muchas pruebas como la prueba de la silueta, KL, Hartigan, Scott, etc y escoge el veredicto que tenga más frecuencia en toda la bateria de pruebas que se hace.

```r
#enlace completo quiere decir que se tomará el máximo de las distancias entre dos pares
#de ítems que pertnecen cada uno a clusters diferentes.

#NbClust(data=df_prom, method = "complete")
```

En este caso arrojó que 2 clusters es lo correcto.

Apreciemos graficamente los clusters. Nota: el porcentaje de los ejes es el porcentaje de explicación de la variación de los datos.

```r
set.seed(14072022) # fija la semilla por reproducibilidad
km1 <- kmeans(df_prom,centers = 2, nstart = 100)
#el porcentaje de los ejes es el porcentaje de explicación de la variación de los datos
#fviz_cluster(km1,
#             data = df_prom,
```

```
#             ellipse.type = "euclid",
#             star.plot = T,
#             repel = T,
#             ggtheme = theme(legend.position = "bottom"))
```

Ahora recuperamos la variable de categorización de cluster para pegarla a la base grande.

```
clusters <- km1$cluster %>% as.data.frame()
colnames(clusters) <- c('tipo_cliente')
df_prom <- cbind(df_prom, clusters)

df_prom <- as.data.frame(df_prom)
df_prom <- add_rownames(df_prom, var = "ID_USER")
df <- df %>% mutate(ID_USER=as.character(ID_USER))
df_prom <- df_prom %>% dplyr::select(ID_USER, tipo_cliente)

df <- df %>% left_join(df_prom, by="ID_USER")
```

Para clasificar a los usuarios usaremos un model logístico sencillo. Usaremos como variable dependiente la dummy de fraude.

```
df_log <- df %>% dplyr::select(monto, hora, establecimiento, ciudad, linea_tc,
                         interes_tc, status_txn, dcto, cashback, device_score,
                         os, d_tarj, d_genero, d_is_prime, d_fraude, tipo_cliente)
```

Imputamos datos missing con un randomforest para no perder observaciones en la regresión. Nota: debido a que tarda mucho se omitió esta linea de código.

```
#library(missForest)
#df_imp <- missForest(df_log)
#df_log <- df_imp$ximp
```

Para no perder observaciones se hará una imputación arbitraria de la variable género y os. Las variabes establecimiento y ciudad no se tomarán en cuenta para el modelo logístico porque casi la mitad de la muestra tiene missing en esas variables.

```
df_log %>% skim()
```

Table 6: Data summary

Name	Piped data
Number of rows	26975
Number of columns	16
Column type frequency:	
factor	4
numeric	12
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
establecimiento	10119	0.62	FALSE	5	Res: 3454, Aba: 3415, Sup: 3402, MPa: 3343
ciudad	11678	0.57	FALSE	4	Tol: 3997, Gua: 3833, Mer: 3761, Mon: 3706
status_txn	0	1.00	FALSE	3	Ace: 18844, En : 5341, Rec: 2790
os	6715	0.75	FALSE	3	%%: 6808, WEB: 6766, AND: 6686

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
monto	0	1.0	499.07	289.31	0.02	246.52	500.50	749.60	999.92	
hora	0	1.0	11.99	6.64	1.00	6.00	12.00	18.00	23.00	
linea_tc	0	1.0	62476.81	21886.89	25000.00	44000.00	62000.00	82000.00	99000.00	
interes_tc	0	1.0	48.22	9.59	32.00	40.00	48.00	57.00	64.00	
dcto	0	1.0	17.47	34.33	0.00	0.00	0.00	18.77	199.36	
cashback	0	1.0	6.26	4.46	0.00	2.79	5.64	8.53	19.99	
device_score	0	1.0	3.00	1.42	1.00	2.00	3.00	4.00	5.00	
d_tarj	0	1.0	0.70	0.46	0.00	0.00	1.00	1.00	1.00	
d_genero	2730	0.9	0.44	0.50	0.00	0.00	0.00	1.00	1.00	
d_is_prime	0	1.0	0.13	0.34	0.00	0.00	0.00	0.00	1.00	
d_fraude	0	1.0	0.03	0.17	0.00	0.00	0.00	0.00	1.00	
tipo_cliente	2	1.0	1.49	0.50	1.00	1.00	1.00	2.00	2.00	

```
df_log$d_genero[is.na(df_log$d_genero)] <- 1#mujer

df_log$os[is.na(df_log$os)] <- "ANDROID"

df_log <- df_log %>% dplyr::select(-c(establecimiento, ciudad))
```

Ajustamos modelo logit con toda la especificación para ver significancia de coeficientes y poder escoger un modelo más chico.

```
modelo_logit <- glm(d_fraude ~ .:. ,family = binomial(link=logit), data = df_log)
summary(modelo_logit)
```

Dividimos base en entrenamiento y prueba.

```
set.seed(14072022)
sample <- sample(c(TRUE, FALSE), nrow(df_log), replace=TRUE, prob=c(0.7,0.3))
train <- df_log[sample, ]
test <- df_log[!sample, ]
```

Nos quedamos con las explicativas que son significativas del modelo y ajustamos modelo a muestra de entrenamiento.

```
modelo_logit_red <- glm(d_fraude ~ monto + d_genero + dcto + d_is_prime + d_tarj +
                          device_score + d_is_prime + status_txn + os + cashback + tipo_cliente +
                          monto:device_score + monto:d_tarj + interes_tc:tipo_cliente +
```

```
                          device_score:tipo_cliente + status_txn:d_genero + dcto:d_tarj +
                          cashback:device_score + cashback:d_tarj + device_score:tipo_cliente +
                          os:d_is_prime + d_genero:d_is_prime,
                        family = binomial(link=logit), data = train)

summary(modelo_logit_red)
```

```
##
## Call:
## glm(formula = d_fraude ~ monto + d_genero + dcto + d_is_prime +
##     d_tarj + device_score + d_is_prime + status_txn + os + cashback +
##     tipo_cliente + monto:device_score + monto:d_tarj + interes_tc:tipo_cliente +
##     device_score:tipo_cliente + status_txn:d_genero + dcto:d_tarj +
##     cashback:device_score + cashback:d_tarj + device_score:tipo_cliente +
##     os:d_is_prime + d_genero:d_is_prime, family = binomial(link = logit),
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.3500  -0.2587  -0.2433  -0.2271   2.8747
##
## Coefficients:
##                                      Estimate        Std. Error z value
## (Intercept)                        -3.7949719         0.3993695  -9.502
## monto                          -1222655.0928618    795501.4871455  -1.537
## d_genero                             0.0255538         0.1072611   0.238
## dcto                           1222655.0919188    795501.4872108   1.537
## d_is_prime                          -0.2796140         0.3077685  -0.909
## d_tarj                               0.1170671         0.1896340   0.617
## device_score                         0.0843601         0.1088995   0.775
## status_txnEn proceso                -0.0014767         0.1537857  -0.010
## status_txnRechazada                 -0.3441228         0.2322172  -1.482
## osANDROID                           -0.1148020         0.1081603  -1.061
## osWEB                               -0.1991591         0.1291311  -1.542
## cashback                         61132754.6460692   39775074.3557063   1.537
## tipo_cliente                         0.4566221         0.2427812   1.881
## monto:device_score                   0.0002375         0.0001817   1.307
## monto:d_tarj                      5219869.3778083   2386595.5910479   2.187
## tipo_cliente:interes_tc             -0.0031466         0.0028323  -1.111
## device_score:tipo_cliente           -0.0767958         0.0604705  -1.270
## d_genero:status_txnEn proceso        0.0768311         0.2141120   0.359
## d_genero:status_txnRechazada         0.1577145         0.3166254   0.498
## dcto:d_tarj                      -5219869.3773907   2386595.5911824  -2.187
## device_score:cashback               -0.0121042         0.0118107  -1.025
## d_tarj:cashback                -460854183.1870724 228499377.8302571  -2.017
## d_is_prime:osANDROID                 0.2017451         0.3422547   0.589
## d_is_prime:osWEB                     0.6440427         0.3675952   1.752
## d_genero:d_is_prime                 -0.1609332         0.2625752  -0.613
##                                 Pr(>|z|)
## (Intercept)                       <2e-16 ***
## monto                             0.1243
## d_genero                          0.8117
## dcto                              0.1243
```

```
## d_is_prime                      0.3636
## d_tarj                          0.5370
## device_score                    0.4385
## status_txnEn proceso            0.9923
## status_txnRechazada             0.1384
## osANDROID                       0.2885
## osWEB                           0.1230
## cashback                        0.1243
## tipo_cliente                    0.0600 .
## monto:device_score              0.1911
## monto:d_tarj                    0.0287 *
## tipo_cliente:interes_tc         0.2666
## device_score:tipo_cliente       0.2041
## d_genero:status_txnEn proceso   0.7197
## d_genero:status_txnRechazada    0.6184
## dcto:d_tarj                     0.0287 *
## device_score:cashback           0.3054
## d_tarj:cashback                 0.0437 *
## d_is_prime:osANDROID            0.5556
## d_is_prime:osWEB                0.0798 .
## d_genero:d_is_prime             0.5399
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5043.5  on 18882  degrees of freedom
## Residual deviance: 5022.7  on 18858  degrees of freedom
##   (2 observations deleted due to missingness)
## AIC: 5072.7
##
## Number of Fisher Scoring iterations: 6
```

```
# Devuelve los valores ajustados, del predictor lineal:
head(predict(modelo_logit_red))
```

```
##         3         4         5         6         7         8
## -3.029215 -3.585830 -3.328199 -3.265038 -3.594642 -3.367649
```

```
# Devuelve las probabilidades ajustadas:
head(predict(modelo_logit_red,type = "response"))
```

```
##          3          4          5          6          7          8
## 0.04612334 0.02696631 0.03461635 0.03679027 0.02673606 0.03332195
```

Hacemos unos pasos intermedios para crear matriz de confusión.

```
predicted <- predict(modelo_logit_red, test, type="response")
```

Encontremos cota óptima de probabilidad para maximizar precisión

```
library(InformationValue)
```

```
##
## Attaching package: 'InformationValue'

## The following objects are masked from 'package:caret':
##
##     confusionMatrix, precision, sensitivity, specificity
```

```
optimal <- optimalCutoff(test$d_fraude, predicted)[1]
```

Creamos matrix de confusión.

```
confusionMatrix(test$d_fraude, predicted)
```

```
##      0   1
## 0 7840 250
```

No hay clasificación de fraude en la muestra de entrenamiento.

Evaluemos nuestra matriz de confusión. Obtendremos las siguientes métricas de nuestra tabla de confusión: sensitivity, specificity y total missclasification rate.

```
sensitivity(test$d_fraude, predicted)
```

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

```
specificity(test$d_fraude, predicted)
```

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

```
misClassError(test$d_fraude, predicted, threshold=optimal)
```

```
## [1] 0.031
```

Tiene un error de misclassification muy bajo por lo que el modelo no es tan malo para predecir.

Conclusiones

La ventaja del modelo de regresión logística es que es sencillo de utilizar y la interpretación de sus coeficientes es simple ya que solamente representan el cambio en la probabilidad de pertenecer al grupo X o no.

Podemos obtener los momios de éxito estimados:

$$\frac{\hat{\mu}(\mathbf{x})}{1-\hat{\mu}(\mathbf{x})} = \exp(b_0)\exp(b_1 x_1)\cdots\exp(b_p x_p)$$

Los exponentes de los coeficientes estimados se llaman *factores de riesgo*.

La desventaja del modelo de clasificación logística es que supone que el costo de clasificación errónea es unitario y las probabilidades iniciales son iguales, lo cual son grandes supuestos y no responden adecuadamente a las necesidades del contexto o de información a priori que se tenga.

Existen mejores modelos como los CART (Classification and Regression Trees) ya que son más flexibles que el modelo logístico por ser modelos no paramétricos de clasificación. También, se pueden usar redes neuronales para clasificar poblaciones sin embargo no domino ese método de clasificación.

Una desventaja del modelo escogido es que puede que esté sobreajustado ya que las pruebas de sensibilidad y especificidad salieron muy cerradas implicando que tal vez no sea bueno prediciendo en nuevas muestras de prueba

El modelo usa como variable significativa la categorización que hice con el algorítmo de k-medias, por lo que da una pauta para creer que sí existen tipos de clientes procilives al uso fraudulento del crédito y que con pocos demográficos y variables explicativas se puede categorizar.

El modelo no es bueno clasificando positivamente los casos fraudulentos pues no hizo ninguna clasificación positiva, no obstante, su error de claisifcación para los casos negativos es muy bajo (del 3%). Por lo que al menos no se equivoca en determinar una transacción no fraudulenta.

Los datos parecen estar balanceados en todos los sentidos ya que no había patrones evidentes de correlación o de estructuras de dependencia entre variables o conjuntos de ellas. Para saber si existiera alguna estructura de dependencia tendría que hacerse un analisis de dependencia con cópulas y simulación pero carecí de tiempo para realizarlo. No obstante, con el análisis exploratorio que realicé no encontré indicios de anomalías en los datos. El muestreo con el que se obtuvo la base parece haber estado bien aleatorizado y tal vez la muestra sea representativa de la población.

Aun así, los clusters fueron claros y a pesar de que no pude anexar al documento la visualización de ellos (la anexo en el repositorio), la gráfica los muestra bien separados con un ligero overlap de observaciones.