Proceso de construcción de modelos

A continuación se detallan brevemente los pasos seguidos en el análisis y construcción de los modelos predictivos.

Utilizamos el Dataset que ha sido preparado para el análisis.

```
df = read.csv("clean.csv",sep=";")
```

Eliminamos algunas variables que no entraran en el análisis:

```
cols = colnames(df)
df <- df[ ,!colnames(df) %in% c("dummy",'X','id')]
```

buscamos si hay alguna columna nula o con valor constante.

```
for (i in cols)
{
  if (  var(df[,i])==0  ) {print (paste("Variable a elminar: ",i))
  df <- df[,!colnames(df)== i ] }
}
```

Particionamos aleatoriamente la muestra en entrenamiento y validación

```
Sample  <- sample.split(df, SplitRatio = .80)
dfTrain <- subset(df, Sample == TRUE)
dfValid  <- subset(df, Sample == FALSE)
```

MODELO DE REGRESIÓN LINEAL

Planteamos un modelo de regresión lineal con el fin de hacer una previsión de la variable target en función del resto de variables y con los datos del dataframe de entrenamiento.

```
reg <- lm(target~., data=dfTrain)
```

Vamos a valorar la capacidad del modelo en terminos de la variable target # como si fuera dicotómica;0 ó 1 si el target es positivo.

Valorarlo en términos de R^2 no parece la mejor opción ya que sus valores son cercanos a cero.

Multiple R-squared: 0.001606, Adjusted R-squared: 0.0005947

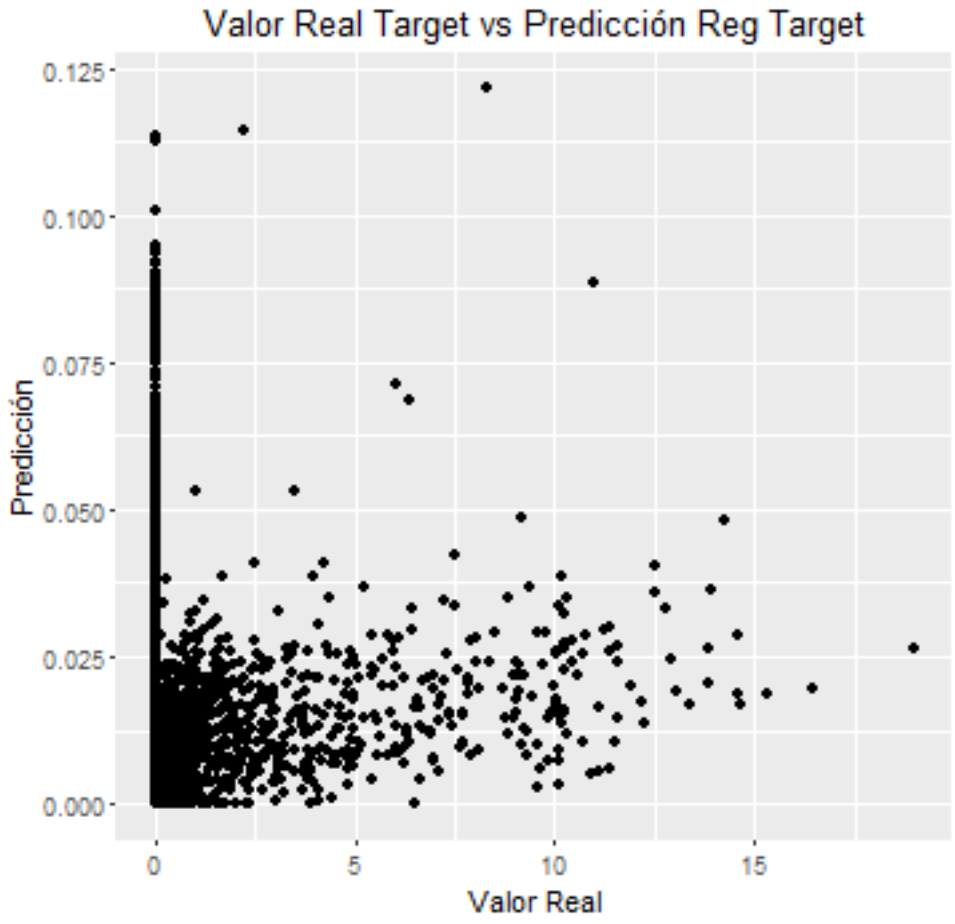
Para transformar la variable en categórica buscaremos un valor que maximice el número de aciertos.

```
maxprecisionreg <-0

for (i in seq(5000))

{

  Precision <- sum(dfreg$target>0 & dfreg$pred>=1/i) +
sum(dfreg$target==0 & dfreg$pred<=1/i)

  ifelse(Precision>maxprecisionreg,maxprecisionreg <- 1/i,0)

}
```

Tenemos por tanto, para éste modelo, los siguientes resultados:

Aciertos (0,1) = 0.2013096

Error medio cometido = 0.01665671


Valor Real Target vs Predicción Reg Target
0.125
0.100
0.075
0.050
0.025
0.000
Predicción
0
5
10
15
Valor Real

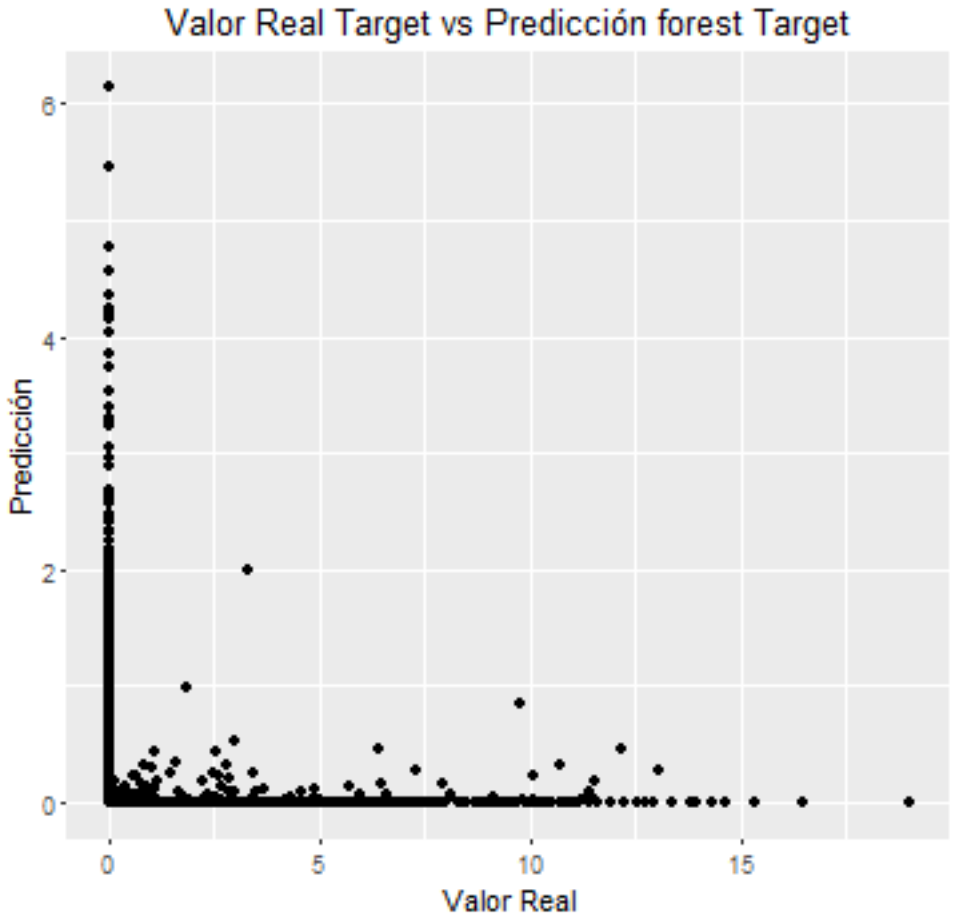
MODELO RANDOM FOREST

Planteamos ahora un modelo Random Forest

Tenemos para éste modelo, los siguientes resultados:

Aciertos (0,1) = 0.7976673

Error medio cometido = 0.01518812



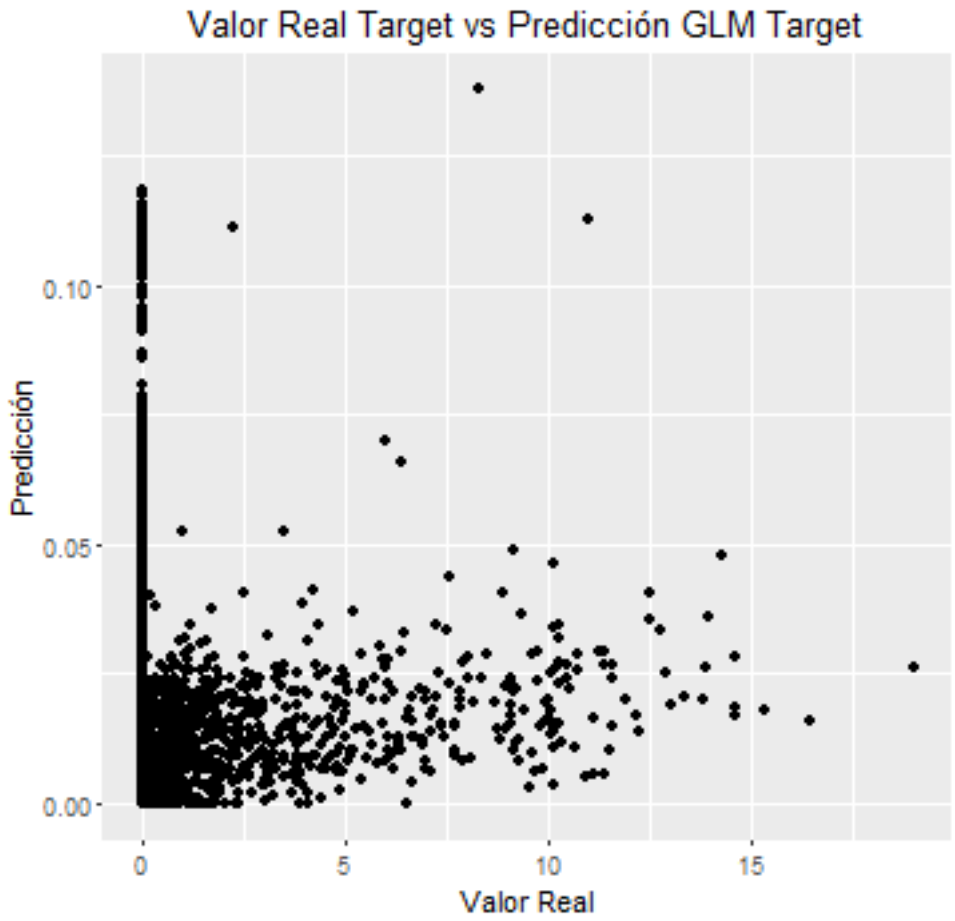
MODELO GLM

Planteamos ahora un modelo GLM

Tenemos para éste modelo, los siguientes resultados:

Aciertos (0,1) = 0.2025096

Error medio cometido = 0.01667306



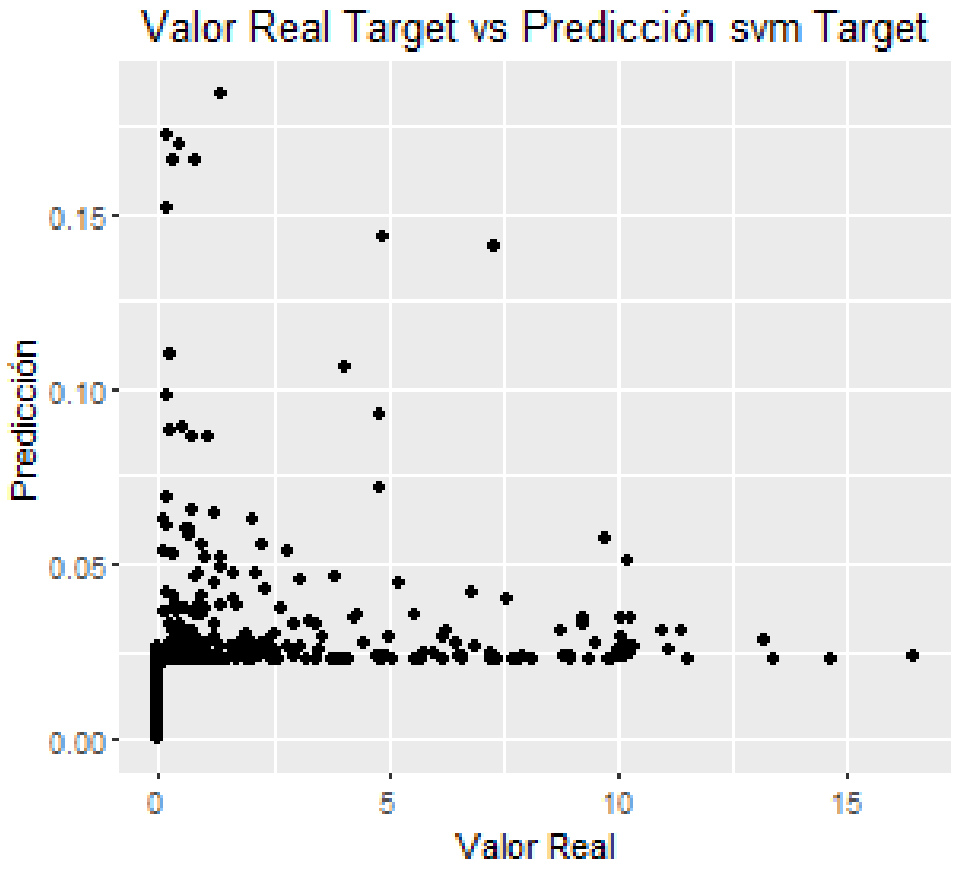
MODELO SVM

Planteamos ahora un modelo SVM

Tenemos para éste modelo, los siguientes resultados:

Aciertos (0,1) = 0.002929293

Error medio cometido = 0.0260049



Llegados hasta aquí, el modelo que mejor predice, tanto en términos de error medio como en términos de (0,1) es el Random Forest.

	LM	RF	GLM	SVM
Aciertos (0,1)	0.2013096	0.7976673	0.2025096	0.002929293
Error medio cometido	0.01665671	0.01518812	0.01667306	0.0260049

Dada la singularidad de la variable target, con un 99% de ceros, se plantea un modelo mixto:

Partiendo del random forest que acierta en un 79% si la variable target es 0 o positiva, proponemos dar un target de previsión de cero a los que el random forest ha clasificado como ceros (según nuestro criterio) y para los positivos plantear un modelo de previsión.

Planteados los cuatro modelos sobre el dataframe de target positivo, éstos son los resultados:

	LM	RF	GLM	SVM
Suma Errores	1745,676	2564,667	1740,348	1428,5

Aplicaremos en una primer etapa el random forest y después el svm para los target positivos,por ser el de menor error.

```
predforest <- predict(forest,dfTrain)

predsvm <- predict(svmpos,newdata=dfTrain)
```

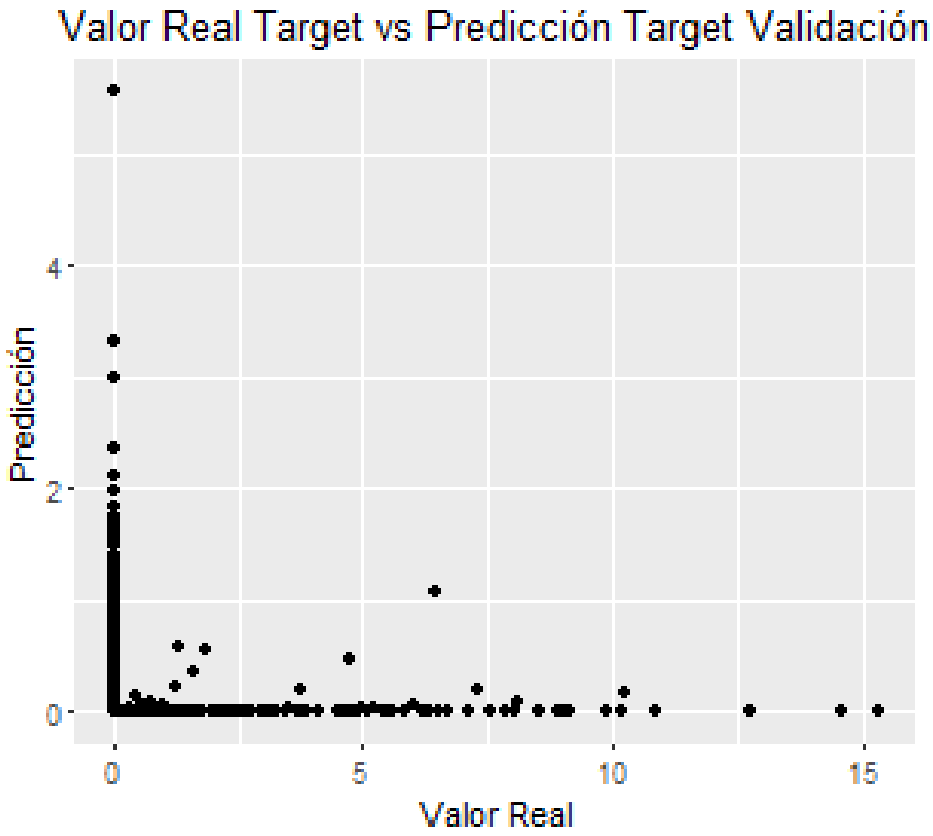
Cálculo de la previsión:

```
prevmixta <- ifelse(predforest<limitforest,0,1) *
ifelse(predsvm<0,0, predsvm)

errormed_prevmixta <- sum(abs(dfTrain$prevmixta-
dfTrain$target))/count(dfTrain)
```

El error medio de éste modelo es 0.499, peor que los modelos anteriores, por lo que aplicamos por tanto el Random Forest al dataframe de validación al ser el mejor modelo de los probados.

predValid <- predict(forest,newdata=dfValid)

errormedvalid <- sum(abs(predValid-dfValid$target))/count(dfValid)



Solo queda aplicar el modelo seleccionado al dataframe test

df = read.csv("test_clean.csv",sep=";")

prevtarget_test <- predict(forest,newdata=df)