TFM - Kaggle House Prices: Advanced Regression Techniques with caret

02 - 01 Ingeniería de características

Juan Carlos Santiago Culebras

2019-09-22

En esta fase se profundiza en el análisis descriptivo del conjunto de datos, y se intentará modificar el conjunto de características para aumentar su eficacia predictiva.

La idea es generar variaciones sobre cómo realizar las transformaciones, tanto en el tipo de transformaciones a aplicar como en el orden en el que se realizan y pasar las diferentes salidas a las siguientes fases de tal forma que se puedan identificar que opciones son las mejores.

Primeros pasos

Librerías

Realizamos la carga de las librerías necesarias

```
if(!is.element("dplyr", installed.packages()[, 1]))
      install.packages("dplyr", repos = 'http://cran.us.r-project.org')
library(dplyr)

if(!is.element("tidyr", installed.packages()[, 1]))
      install.packages("tidyr", repos = 'http://cran.us.r-project.org')
library(tidyr)

if(!is.element("ggplot2", installed.packages()[, 1]))
      install.packages("ggplot2", repos = 'http://cran.us.r-project.org')
library(ggplot2)

if(!is.element("tibble", installed.packages()[, 1]))
      install.packages("tibble", repos = 'http://cran.us.r-project.org')
library(tibble)

# grid.arrange / marrangeGrob
library(gridExtra)

# correlation matrixes - rcorr (niveles de significación)
if(!is.element("Hmisc", installed.packages()[, 1]))
      install.packages("Hmisc", repos = 'http://cran.us.r-project.org')
library(Hmisc)

# correlation matrixes - ggcorr
if(!is.element("GGally", installed.packages()[, 1]))
      install.packages("GGally", repos = 'http://cran.us.r-project.org')
library(GGally)
```

```r
# correlation matrixes - corrplot
if(!is.element("corrplot", installed.packages()[, 1]))
      install.packages("corrplot", repos = 'http://cran.us.r-project.org')
library(corrplot)


if(!is.element("caret", installed.packages()[, 1]))
      install.packages("caret", repos = 'http://cran.us.r-project.org')
library(caret)

if(!is.element("fastDummies", installed.packages()[, 1]))
      install.packages("fastDummies", repos = 'http://cran.us.r-project.org')
library(fastDummies)

if(!is.element("devtools", installed.packages()[, 1]))
      install.packages("devtools", repos = 'http://cran.us.r-project.org')
library(devtools)

if(!is.element("ggbiplot", installed.packages()[, 1]))
      devtools::install_github("richardjtelford/ggbiplot", ref = "experimental")
library(ggbiplot)
```

Funciones

```r
ggplotHistogramaDensidad <- function (strCampo,ds) {
  require(psych)
  require(ggplot2)

  # Estudio mas detallado con curtosis y sesgo
  strDescribe = paste("d <- psych::describe(ds$",strCampo,")",sep = "")
  eval(parse(text = paste(strDescribe)))

  title <- strCampo
  subtitle <- paste("Kurtosis:", signif(d$kurtosis,3)
                    ," / Skew:", signif(d$skew,3))

  p <- ggplot(ds, aes(x=get(strCampo))) +
    geom_histogram(aes(y=..density..), colour="black", fill="white") +
    geom_vline(aes(xintercept=mean(get(strCampo))),
            color="blue", linetype="dashed", size=1) +
    geom_density(alpha=.2, fill="#FF6666") +
    #scale_x_continuous(breaks = 100000) +
    labs(title=title, subtitle=subtitle, x = strCampo) +
    theme(plot.subtitle = element_text(size=10)) +
    scale_x_continuous(labels = scales::comma)

  return(p)
}
```

Cargamos datos

Partimos del dataset generado en la etapa anterior fichero F01_dsDataAll (datos limpios)

Se ha realizado: *Verificación de datos y campos* Tratamiento de valores perdidos *Tratamiento de outliers

```
load('./F01_Datos/F01_dsDataAll.RData')

# Definición de campos
dsCampos <- read.csv("./input/campos.csv",sep=";",stringsAsFactors = FALSE)

dsCampos <- dsCampos %>%
  mutate_if(is.factor, as.character)

# Dejamos el Tipo y el segmento como factor
dsCampos$Tipo <- as.factor(dsCampos$Tipo)
dsCampos$Segmento <- as.factor(dsCampos$Segmento)
```

Ingeniería de características

Normalización

En el análisis inicial se detectó que la mayoría de las variables continuas están sesgadas por lo que he decido aplicar la función log a todas ellas. Para ello se crea una función que convierte una variable del dataset, pasada como parámetro aplicándole la función log, esta función se llama mediante apply sobre todos los campos continuos.

```
# Selecciono variable continuas
dsCamposContinua <- dsCampos %>%
  filter(Tipo=="Continua")  %>%
  select(Campo)

# creo una función para aplicar la función log sobre variables del dataset
funcMutateLog <- function(var){
  dsDataAll[,var] <- log(dsDataAll[,var])
  assign('dsDataAll',dsDataAll,envir=.GlobalEnv) # Pendiente buscar solución mas elegante
}

apply(dsCamposContinua, MARGIN=1, funcMutateLog)
```

Presento resultados de la normalización

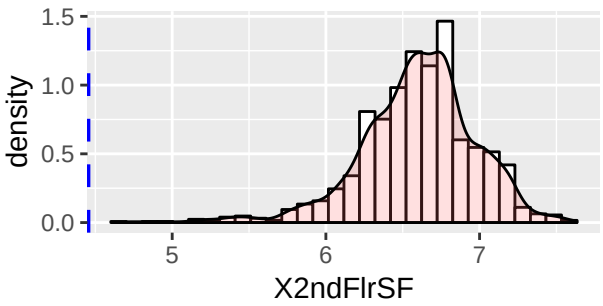
```
gs <- apply(dsCamposContinua, MARGIN=1, ggplotHistogramaDensidad, ds=dsDataAll)
marrangeGrob(grobs=gs, nrow=2, ncol=2)
```

SalePrice

Kurtosis: 0.793 / Skew: 0.121

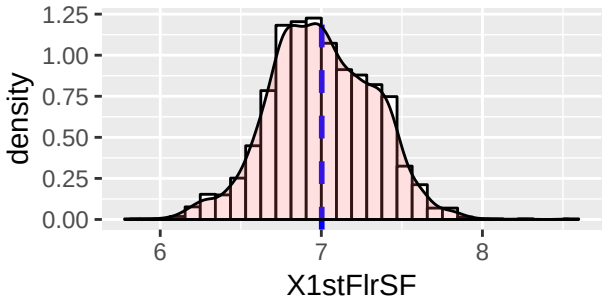
X2ndFlrSF

Kurtosis: NaN / Skew: NaN



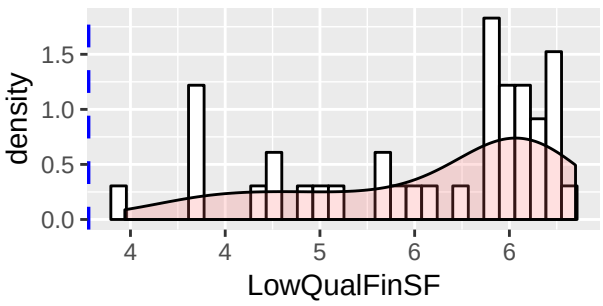
X1stFlrSF

Kurtosis: 0.04 / Skew: 0.0294



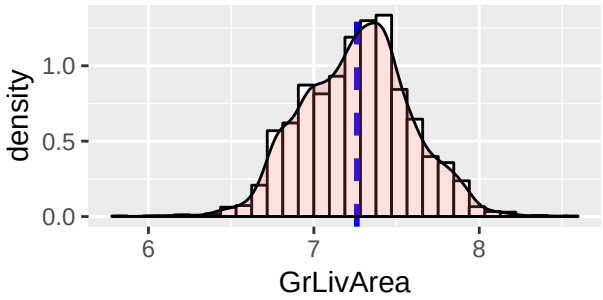
LowQualFinSF

Kurtosis: NaN / Skew: NaN



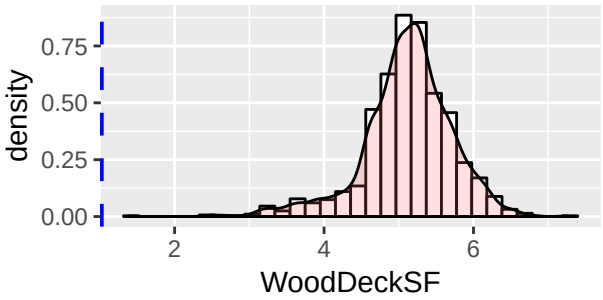
GrLivArea

Kurtosis: 0.0998 / Skew: −0.0228



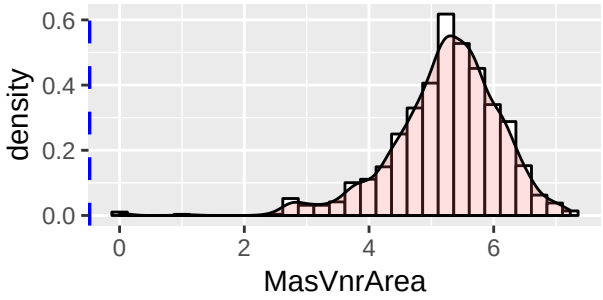
WoodDeckSF

Kurtosis: NaN / Skew: NaN



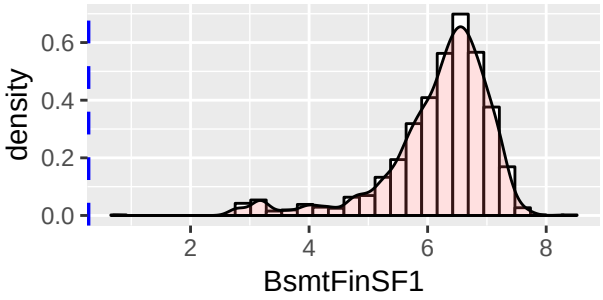
MasVnrArea

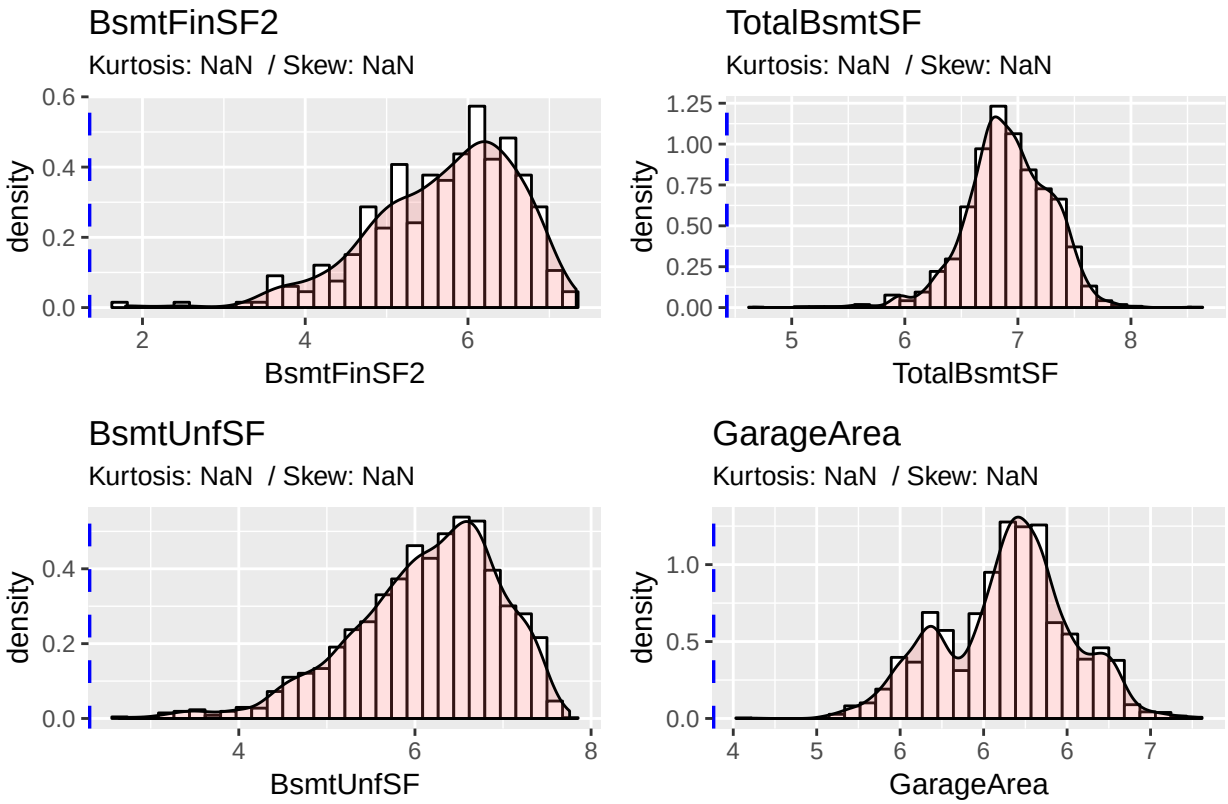
Kurtosis: NaN / Skew: NaN



BsmtFinSF1

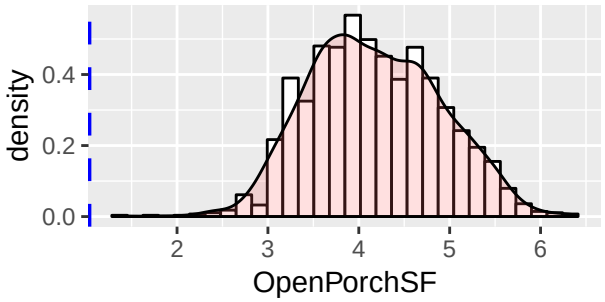
Kurtosis: NaN / Skew: NaN




BsmtFinSF2
Kurtosis: NaN / Skew: NaN
density
BsmtFinSF2
TotalBsmtSF
Kurtosis: NaN / Skew: NaN
density
TotalBsmtSF
BsmtUnfSF
Kurtosis: NaN / Skew: NaN
density
BsmtUnfSF
GarageArea
Kurtosis: NaN / Skew: NaN
density
GarageArea

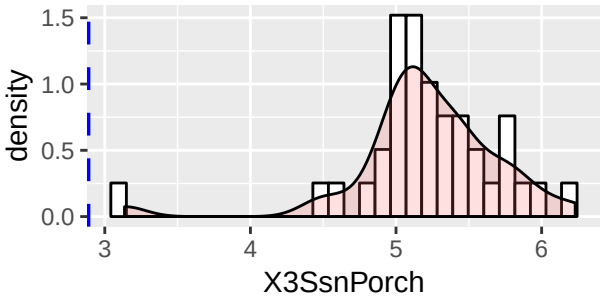
OpenPorchSF

Kurtosis: NaN / Skew: NaN



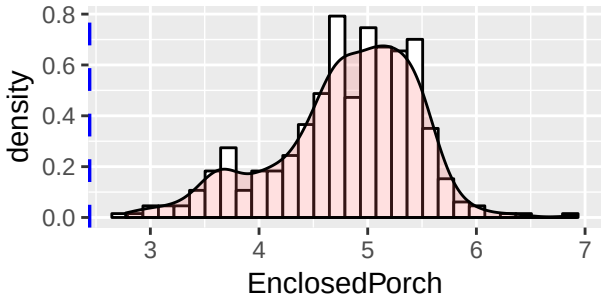
X3SsnPorch

Kurtosis: NaN / Skew: NaN



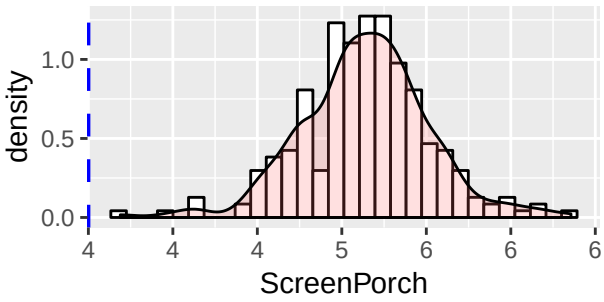
EnclosedPorch

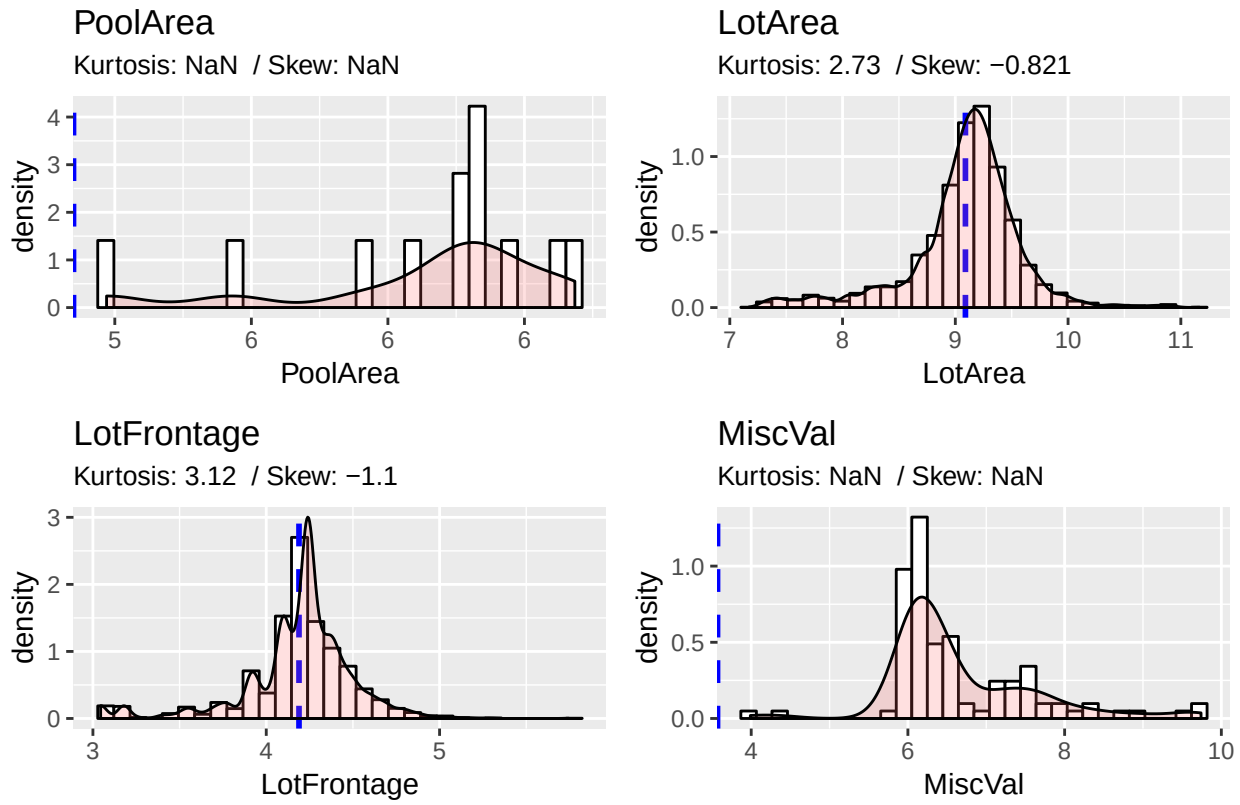
Kurtosis: NaN / Skew: NaN



ScreenPorch

Kurtosis: NaN / Skew: NaN






```
dsDataAll[mapply(is.infinite, dsDataAll)] <- 0

rm(gs)
rm(funcMutateLog)
```

Salvar progreso

```
#save(dsDataAll, file = './FO2_Datos/FO2_01_dsDataAll.RData')
# load('./FO2_Datos/FO2_01_dsDataAll.RData')
```

Tratamiento de variables nominales - dummy

Todas las variables nominales se convertirán a numéricas binarias, para ello cada variable generara nuevas variables una por cada valor existente en el conjunto de datos, indicando como valor 0 o 1, ausencia o presencia del valor.

Existen multitud de formas en r de realizar esta transformación, yo he seleccionado usar dummy_cols del paquete fastDummies.

```
# Obtengo campos originales una vez trasformados
dsCamposActuales <- data.frame(unlist(sapply(dsDataAll, class))) %>%
  select(Tipo = 1) %>%
  rownames_to_column("Campo")

CamposFactor <- filter(dsCamposActuales, Tipo == "factor" & Campo != "indTrain") %>% select(Campo)
```

```
dsDummy <- dsDataAll %>% select(c("Id",c(CamposFactor$Campo)))
dsDummy <- fastDummies::dummy_cols(dsDummy)
dsDummy <- select(dsDummy,-c(CamposFactor$Campo))

dsDataAll <- select(dsDataAll,-c(CamposFactor$Campo))

dsDataAll <- dsDataAll %>%
    inner_join(dsDummy, by="Id")

rm(CamposFactor)
rm(dsDummy)
```

Salvar progreso

```
#save(dsDataAll, file = './F02_Datos/F02_01_dsDataAll.RData')
# load('./F02_Datos/F02_01_dsDataAll.RData')
```

Eliminación de variables con varianza próxima a cero

Si una variable tiene casi todas las observaciones con un mismo valor, su varianza será próxima a cero. Estas variables pueden añadir mas ruido que información, también dan problemas cuando se seleccionan los conjuntos de entrenamiento ya que si en la variable solo queda un valor puede producir que el entrenamiento sea erróneo.

He utilizado la función nearZeroVar() del paquete caret para seleccionar estas variables y se han eliminado del dataset general.

La eliminación de estas variables se debería realizar antes de la normalización, ya que después ya no tendrán la varianza cero, pero en este caso como solo se han normalizado las variables continuas, se ha decidido hacer posteriormente y se ha verificado que el resultado no empeoraba en la siguiente fase.

```
dsVarianzaCero <- dsDataAll %>%
  nearZeroVar(saveMetrics = TRUE) %>%
  rownames_to_column(var = "Campo") %>%
  filter(zeroVar==TRUE | nzv==TRUE)

dsDataAll <- dsDataAll %>%
    select(-c(dsVarianzaCero$Campo))

rm(dsVarianzaCero)
```

Salvar progreso

```
#save(dsDataAll, file = './F02_Datos/F02_01_dsDataAll.RData')
# load('./F02_Datos/F02_01_dsDataAll.RData')
```

Elimino objetos que no se van a usar

```
rm(dsCampos)
rm(dsCamposContinua)
rm(dsCamposActuales)
rm(dsCamposOriginales)
```

Análisis de Componentes Principales

Redución de la dimensionalidad utilizando el Análisis de Componentes Principales.

El objetivo del PCA es buscar una representación alternativa y reducida de las tuplas originales formadas por n atributos. Se buscan los k vectores ortogonales (k < n) que mejor representan los n atributos originales.

Como el PCA no es invariante de escala, es recomendable estandarizar las variables antes de aplicarlo.

En el cálcuo de componentes principales se escluye la variable respuesta.

```
# Selecciono las variables que forman el problema,
# eliminando la variable objetivo SalePrice, el Id y el indTrain
dsDataAllPCA <- dsDataAll[,-c(1:3)]
```

Cálculo de componentes principales.

Utilizo todos los datos disponibles ya que no esta relacionado con la variable objetivo

```
pca <- prcomp(dsDataAllPCA, center = TRUE, scale = TRUE)
```

Realizamos estudio y seleccionamos componentes principales

```
#ggbiplot(pca)

#summary(pca)

# Se guarda la proporción de varianza explicada y acumulada de los componentes
summaryPCA <- as.data.frame(summary(pca)$importance) %>%
    rownames_to_column("Estadistico") %>%
    gather(PC, Valor, PC1:PC92) %>%
    mutate(PC = as.integer(gsub("PC","",PC)))

filter(summaryPCA, Estadistico == 'Cumulative Proportion') # & Valor < 0.99)
```

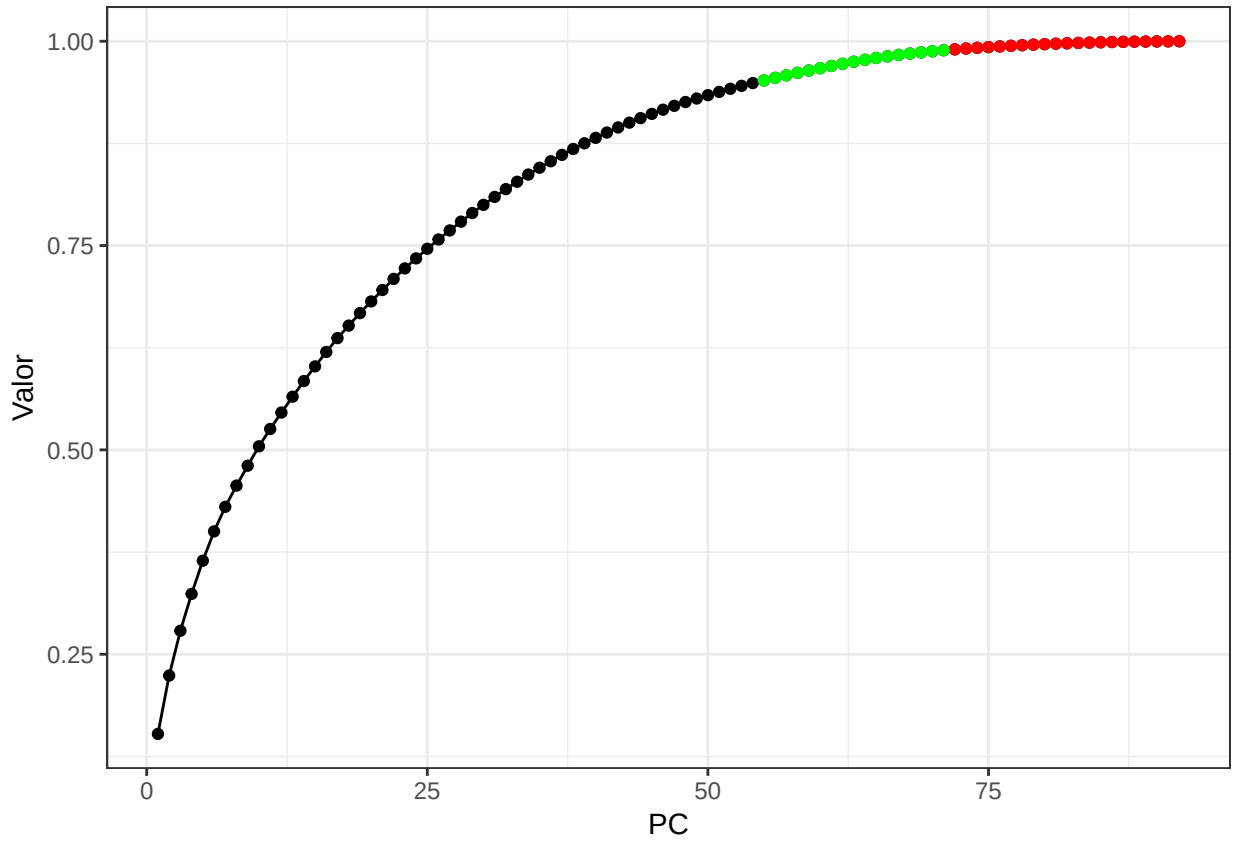
```
##              Estadistico PC   Valor
## 1  Cumulative Proportion  1 0.15250
## 2  Cumulative Proportion  2 0.22391
## 3  Cumulative Proportion  3 0.27871
## 4  Cumulative Proportion  4 0.32383
## 5  Cumulative Proportion  5 0.36443
## 6  Cumulative Proportion  6 0.40028
## 7  Cumulative Proportion  7 0.43037
## 8  Cumulative Proportion  8 0.45630
## 9  Cumulative Proportion  9 0.48069
## 10 Cumulative Proportion 10 0.50441
## 11 Cumulative Proportion 11 0.52561
## 12 Cumulative Proportion 12 0.54570
## 13 Cumulative Proportion 13 0.56513
## 14 Cumulative Proportion 14 0.58422
## 15 Cumulative Proportion 15 0.60219
## 16 Cumulative Proportion 16 0.61984
## 17 Cumulative Proportion 17 0.63680
```

```
## 18 Cumulative Proportion 18 0.65215
## 19 Cumulative Proportion 19 0.66739
## 20 Cumulative Proportion 20 0.68168
## 21 Cumulative Proportion 21 0.69564
## 22 Cumulative Proportion 22 0.70920
## 23 Cumulative Proportion 23 0.72205
## 24 Cumulative Proportion 24 0.73431
## 25 Cumulative Proportion 25 0.74606
## 26 Cumulative Proportion 26 0.75746
## 27 Cumulative Proportion 27 0.76852
## 28 Cumulative Proportion 28 0.77929
## 29 Cumulative Proportion 29 0.78983
## 30 Cumulative Proportion 30 0.79981
## 31 Cumulative Proportion 31 0.80953
## 32 Cumulative Proportion 32 0.81907
## 33 Cumulative Proportion 33 0.82809
## 34 Cumulative Proportion 34 0.83689
## 35 Cumulative Proportion 35 0.84521
## 36 Cumulative Proportion 36 0.85317
## 37 Cumulative Proportion 37 0.86094
## 38 Cumulative Proportion 38 0.86819
## 39 Cumulative Proportion 39 0.87511
## 40 Cumulative Proportion 40 0.88176
## 41 Cumulative Proportion 41 0.88829
## 42 Cumulative Proportion 42 0.89452
## 43 Cumulative Proportion 43 0.90037
## 44 Cumulative Proportion 44 0.90591
## 45 Cumulative Proportion 45 0.91109
## 46 Cumulative Proportion 46 0.91623
## 47 Cumulative Proportion 47 0.92100
## 48 Cumulative Proportion 48 0.92564
## 49 Cumulative Proportion 49 0.92992
## 50 Cumulative Proportion 50 0.93405
## 51 Cumulative Proportion 51 0.93802
## 52 Cumulative Proportion 52 0.94183
## 53 Cumulative Proportion 53 0.94545
## 54 Cumulative Proportion 54 0.94882
## 55 Cumulative Proportion 55 0.95208
## 56 Cumulative Proportion 56 0.95522
## 57 Cumulative Proportion 57 0.95833
## 58 Cumulative Proportion 58 0.96135
## 59 Cumulative Proportion 59 0.96423
## 60 Cumulative Proportion 60 0.96700
## 61 Cumulative Proportion 61 0.96975
## 62 Cumulative Proportion 62 0.97238
## 63 Cumulative Proportion 63 0.97491
## 64 Cumulative Proportion 64 0.97733
## 65 Cumulative Proportion 65 0.97954
## 66 Cumulative Proportion 66 0.98151
## 67 Cumulative Proportion 67 0.98326
## 68 Cumulative Proportion 68 0.98495
## 69 Cumulative Proportion 69 0.98636
## 70 Cumulative Proportion 70 0.98774
## 71 Cumulative Proportion 71 0.98897
```

```
## 72 Cumulative Proportion 72 0.99007
## 73 Cumulative Proportion 73 0.99106
## 74 Cumulative Proportion 74 0.99195
## 75 Cumulative Proportion 75 0.99279
## 76 Cumulative Proportion 76 0.99360
## 77 Cumulative Proportion 77 0.99439
## 78 Cumulative Proportion 78 0.99511
## 79 Cumulative Proportion 79 0.99579
## 80 Cumulative Proportion 80 0.99640
## 81 Cumulative Proportion 81 0.99695
## 82 Cumulative Proportion 82 0.99748
## 83 Cumulative Proportion 83 0.99791
## 84 Cumulative Proportion 84 0.99832
## 85 Cumulative Proportion 85 0.99870
## 86 Cumulative Proportion 86 0.99902
## 87 Cumulative Proportion 87 0.99923
## 88 Cumulative Proportion 88 0.99943
## 89 Cumulative Proportion 89 0.99961
## 90 Cumulative Proportion 90 0.99975
## 91 Cumulative Proportion 91 0.99989
## 92 Cumulative Proportion 92 1.00000
```

```r
g <- filter(summaryPCA, Estadistico == 'Cumulative Proportion')

ggplot(data = g, aes(x = PC, y = Valor)) +
  geom_line() +
  geom_point() +
  geom_point(data = filter(g,Valor>0.99),
             color = "red") +
  geom_point(data = filter(g,Valor<0.99 & Valor>0.95),
             color = "green") +
  theme_bw()
```



Seleccionamos los componentes cuyo valor esplica el 95 % de la varianza

```
# Seleccionamos los componentes cuyo valor esplica el 95 % de la varianza
filter(summaryPCA, Estadistico == 'Cumulative Proportion', Valor < 0.95)
```

```
##              Estadistico PC   Valor
## 1  Cumulative Proportion  1 0.15250
## 2  Cumulative Proportion  2 0.22391
## 3  Cumulative Proportion  3 0.27871
## 4  Cumulative Proportion  4 0.32383
## 5  Cumulative Proportion  5 0.36443
## 6  Cumulative Proportion  6 0.40028
## 7  Cumulative Proportion  7 0.43037
## 8  Cumulative Proportion  8 0.45630
## 9  Cumulative Proportion  9 0.48069
## 10 Cumulative Proportion 10 0.50441
## 11 Cumulative Proportion 11 0.52561
## 12 Cumulative Proportion 12 0.54570
## 13 Cumulative Proportion 13 0.56513
## 14 Cumulative Proportion 14 0.58422
## 15 Cumulative Proportion 15 0.60219
## 16 Cumulative Proportion 16 0.61984
## 17 Cumulative Proportion 17 0.63680
## 18 Cumulative Proportion 18 0.65215
## 19 Cumulative Proportion 19 0.66739
## 20 Cumulative Proportion 20 0.68168
```

```
## 21 Cumulative Proportion 21 0.69564
## 22 Cumulative Proportion 22 0.70920
## 23 Cumulative Proportion 23 0.72205
## 24 Cumulative Proportion 24 0.73431
## 25 Cumulative Proportion 25 0.74606
## 26 Cumulative Proportion 26 0.75746
## 27 Cumulative Proportion 27 0.76852
## 28 Cumulative Proportion 28 0.77929
## 29 Cumulative Proportion 29 0.78983
## 30 Cumulative Proportion 30 0.79981
## 31 Cumulative Proportion 31 0.80953
## 32 Cumulative Proportion 32 0.81907
## 33 Cumulative Proportion 33 0.82809
## 34 Cumulative Proportion 34 0.83689
## 35 Cumulative Proportion 35 0.84521
## 36 Cumulative Proportion 36 0.85317
## 37 Cumulative Proportion 37 0.86094
## 38 Cumulative Proportion 38 0.86819
## 39 Cumulative Proportion 39 0.87511
## 40 Cumulative Proportion 40 0.88176
## 41 Cumulative Proportion 41 0.88829
## 42 Cumulative Proportion 42 0.89452
## 43 Cumulative Proportion 43 0.90037
## 44 Cumulative Proportion 44 0.90591
## 45 Cumulative Proportion 45 0.91109
## 46 Cumulative Proportion 46 0.91623
## 47 Cumulative Proportion 47 0.92100
## 48 Cumulative Proportion 48 0.92564
## 49 Cumulative Proportion 49 0.92992
## 50 Cumulative Proportion 50 0.93405
## 51 Cumulative Proportion 51 0.93802
## 52 Cumulative Proportion 52 0.94183
## 53 Cumulative Proportion 53 0.94545
## 54 Cumulative Proportion 54 0.94882
```

```
## 55 Cumulative Proportion PC55 0.99064
filter(summaryPCA, Estadistico == 'Cumulative Proportion', PC == 55)
```

```
##             Estadistico PC   Valor
## 1 Cumulative Proportion 55 0.95208
```

Guardamos el dataset con los componentes principales

```
# Aplicamos al conjunto para obtener las nuevas variables
dsPCA <- as.data.frame(predict(pca, newdata = dsDataAllPCA))

dsPCA <- dsPCA[,1:55]

#Guardamos en el dataset dsDataAll los resultados
dsDataAll <- cbind(dsDataAll[,1:3], dsPCA)

save(dsDataAll, file = './F02_Datos/F02_02_dsDataAll_PCA.RData')
```