GAMBikes

Beltran

22/10/2019

TO DO LIST

- READ DATA
- SUMMARISE DATA
- GAM

LIBRARIES & SEED

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(knitr)
library(ISLR)
library(boot)
library(splines)
library(ggplot2)
library(gam)
```

```
## Loading required package: foreach

## Loaded gam 1.16.1
```

```
library(skimr)
```

```
##
## Attaching package: 'skimr'

## The following object is masked from 'package:knitr':
##
##     kable
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(car)
```

```
## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:boot':
##
##     logit

## The following object is masked from 'package:dplyr':
##
##     recode
```

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
set.seed(123)
```

ANALISIS EXPLORATORIO

READ DATA

```
bikes <- read.csv("day.csv")
```

SUMMARISE DATA

Comprobamos la distribución de las variables a través de la representación de su histrograma. También observo estadísticos tales como los cuartiles, media, mediana y desviación típica.

```
skim(bikes)
```

```
## Skim summary statistics
##  n obs: 731
##  n variables: 16
##
## -- Variable type:factor -------------------------------------------------------
##  variable missing complete   n n_unique                    top_counts
##    dteday       0      731 731      731 201: 1, 201: 1, 201: 1, 201: 1
##  ordered
##    FALSE
```

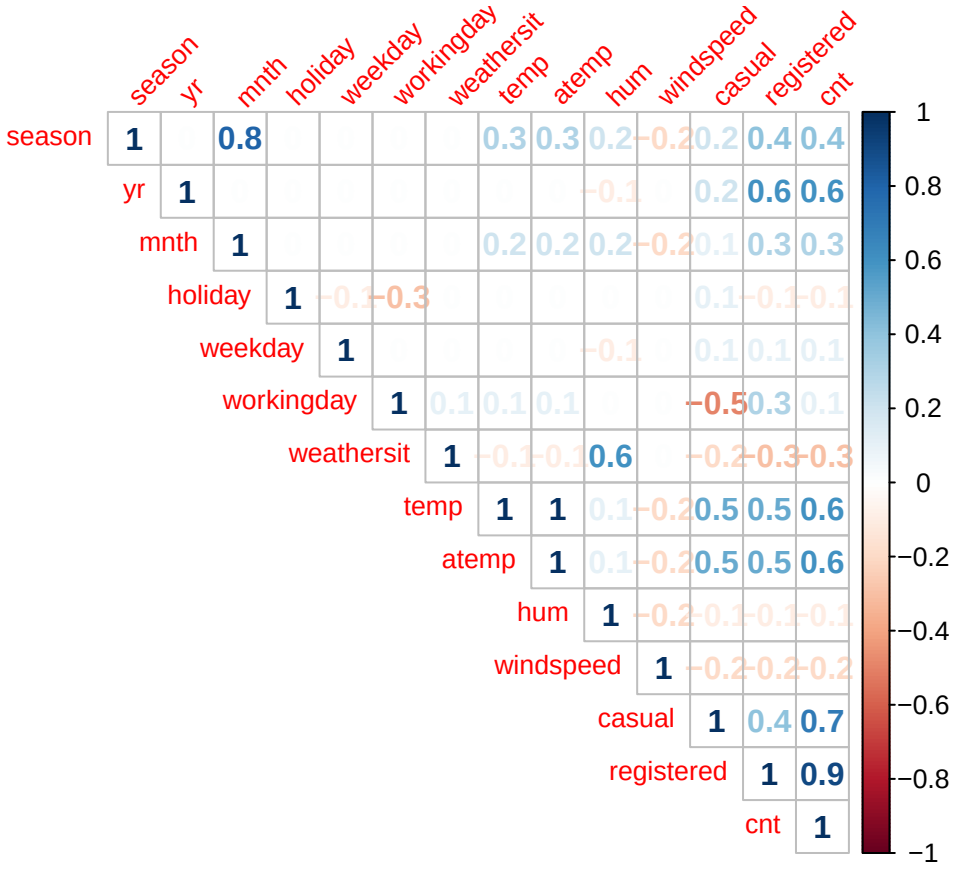
```
##
## -- Variable type:integer -------------------------------------------------------
##    variable missing complete   n     mean      sd p0    p25  p50    p75
##      casual       0      731 731  848.18   686.62  2  315.5  713 1096
##         cnt       0      731 731 4504.35  1937.21 22 3152   4548 5956
##     holiday       0      731 731    0.029    0.17  0    0      0    0
##     instant       0      731 731  366      211.17  1  183.5  366  548.5
##        mnth       0      731 731    6.52     3.45  1    4      7   10
##  registered       0      731 731 3656.17  1560.26 20 2497   3662 4776.5
##      season       0      731 731    2.5      1.11  1    2      3    3
##  weathersit       0      731 731    1.4      0.54  1    1      1    2
##     weekday       0      731 731    3        2     0    1      3    5
##  workingday       0      731 731    0.68     0.47  0    0      1    1
##          yr       0      731 731    0.5      0.5   0    0      1    1
##  p100     hist
##  3410 <U+2587><U+2587><U+2585><U+2582><U+2581><U+2581><U+2581><U+2581>
##  8714 <U+2582><U+2585><U+2585><U+2587><U+2587><U+2585><U+2585><U+2582>
##     1 <U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581>
##   731 <U+2587><U+2587><U+2587><U+2587><U+2587><U+2587><U+2587><U+2587>
##    12 <U+2587><U+2585><U+2587><U+2583><U+2585><U+2587><U+2585><U+2587>
##  6946 <U+2581><U+2585><U+2585><U+2586><U+2587><U+2585><U+2583><U+2583>
##     4 <U+2587><U+2581><U+2587><U+2581><U+2581><U+2587><U+2581><U+2587>
##     3 <U+2587><U+2581><U+2581><U+2585><U+2581><U+2581><U+2581><U+2581>
##     6 <U+2587><U+2587><U+2587><U+2587><U+2581><U+2587><U+2587><U+2587>
##     1 <U+2583><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2587>
##     1 <U+2587><U+2581><U+2581><U+2581><U+2581><U+2581><U+2581><U+2587>
##
## -- Variable type:numeric -------------------------------------------------------
##   variable missing complete   n mean    sd    p0  p25  p50  p75 p100
##      atemp       0      731 731 0.47 0.16  0.079 0.34 0.49 0.61 0.84
##        hum       0      731 731 0.63 0.14  0     0.52 0.63 0.73 0.97
##       temp       0      731 731 0.5  0.18  0.059 0.34 0.5  0.66 0.86
##  windspeed       0      731 731 0.19 0.077 0.022 0.13 0.18 0.23 0.51
##      hist
##  <U+2581><U+2585><U+2587><U+2587><U+2587><U+2587><U+2586><U+2581>
##  <U+2581><U+2581><U+2581><U+2583><U+2587><U+2587><U+2585><U+2582>
##  <U+2581><U+2583><U+2587><U+2586><U+2586><U+2587><U+2587><U+2582>
##  <U+2582><U+2586><U+2587><U+2586><U+2582><U+2581><U+2581><U+2581>
```

Observo que disponemos de un data set con 731 observaciones correpsondientes a 16 variables. No existen valores omitidos en ninguna de las variables. La variable 'instant' muestra únicamente el índice de cada uno de los registros, por tanto, es redundante ya que R de por sí ya indeza los registros. Procedo a eliminarla.

```
bikes <- select(bikes, -instant)
```

Ahora el data set presenta un total de 15 variables.

```
corrplot(round(cor(bikes %>% select_at(vars(-dteday))), 1), method = "number", type = "upper",
         tl.cex = 0.85, tl.srt = 45)
```



Se comprueba que las variables "casual" y "registered" presentan una alta correlación con la variable "cnt", esto se debo a que ambas forman parte de la última puesto que "cnt" resulta de la suma de "casual" y "registered". Para el posterior análisis no se tendrán en cuenta ninguna de las dos, ni "casual", ni "registered".

PREDICTION

MULTIPLE PREDICTORS: GAM

DOF

Calculo los grados de libertad de casa una de las variables numéricas. Estos serán los grados de libertad óptimos. El cálculo se realiza mediante cross-validation.

```
DOFtemp <- smooth.spline(bikes$temp, bikes$cnt, cv = TRUE) #DOF of temp
DOFatemp <- smooth.spline(bikes$atemp, bikes$cnt, cv = TRUE) #DOF of atemp
DOFhum <- smooth.spline(bikes$hum, bikes$cnt, cv = TRUE) #DOF of hum
DOFwindspeed <- smooth.spline(bikes$windspeed, bikes$cnt, cv = TRUE) #DOF of windspeed

DOFtemp; DOFatemp; DOFhum; DOFwindspeed
```

```
## Call:
## smooth.spline(x = bikes$temp, y = bikes$cnt, cv = TRUE)
##
## Smoothing Parameter  spar= 0.8750699  lambda= 0.002094182 (12 iterations)
```

```
## Equivalent Degrees of Freedom (Df): 9.103704
## Penalized Criterion (RSS): 1016814001
## PRESS(l.o.o. CV): 2020811
```

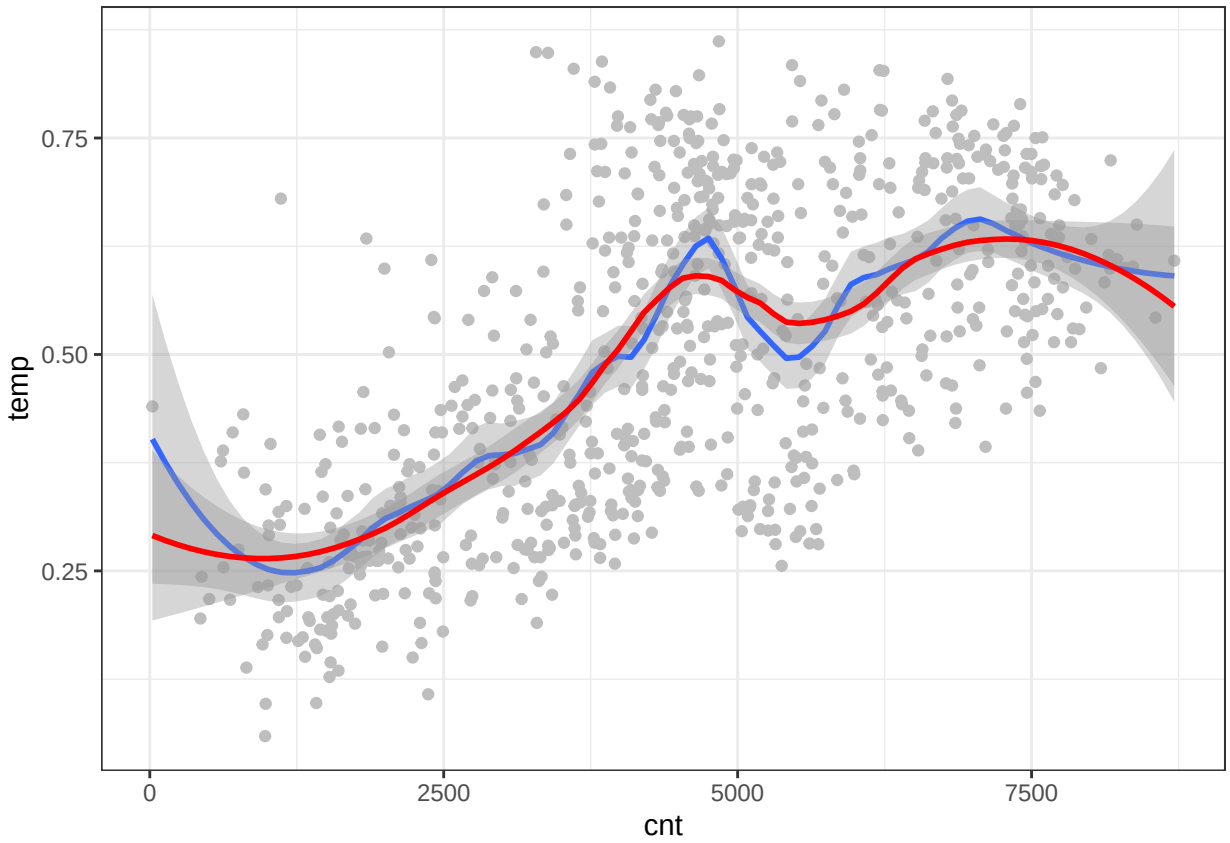
```
## Call:
## smooth.spline(x = bikes$atemp, y = bikes$cnt, cv = TRUE)
##
## Smoothing Parameter  spar= 0.9259008  lambda= 0.00235741 (10 iterations)
## Equivalent Degrees of Freedom (Df): 8.805497
## Penalized Criterion (RSS): 1357061571
## PRESS(l.o.o. CV): 2019174
```

```
## Call:
## smooth.spline(x = bikes$hum, y = bikes$cnt, cv = TRUE)
##
## Smoothing Parameter  spar= 1.126702  lambda= 0.02912609 (14 iterations)
## Equivalent Degrees of Freedom (Df): 4.548876
## Penalized Criterion (RSS): 2086803858
## PRESS(l.o.o. CV): 3465906
```

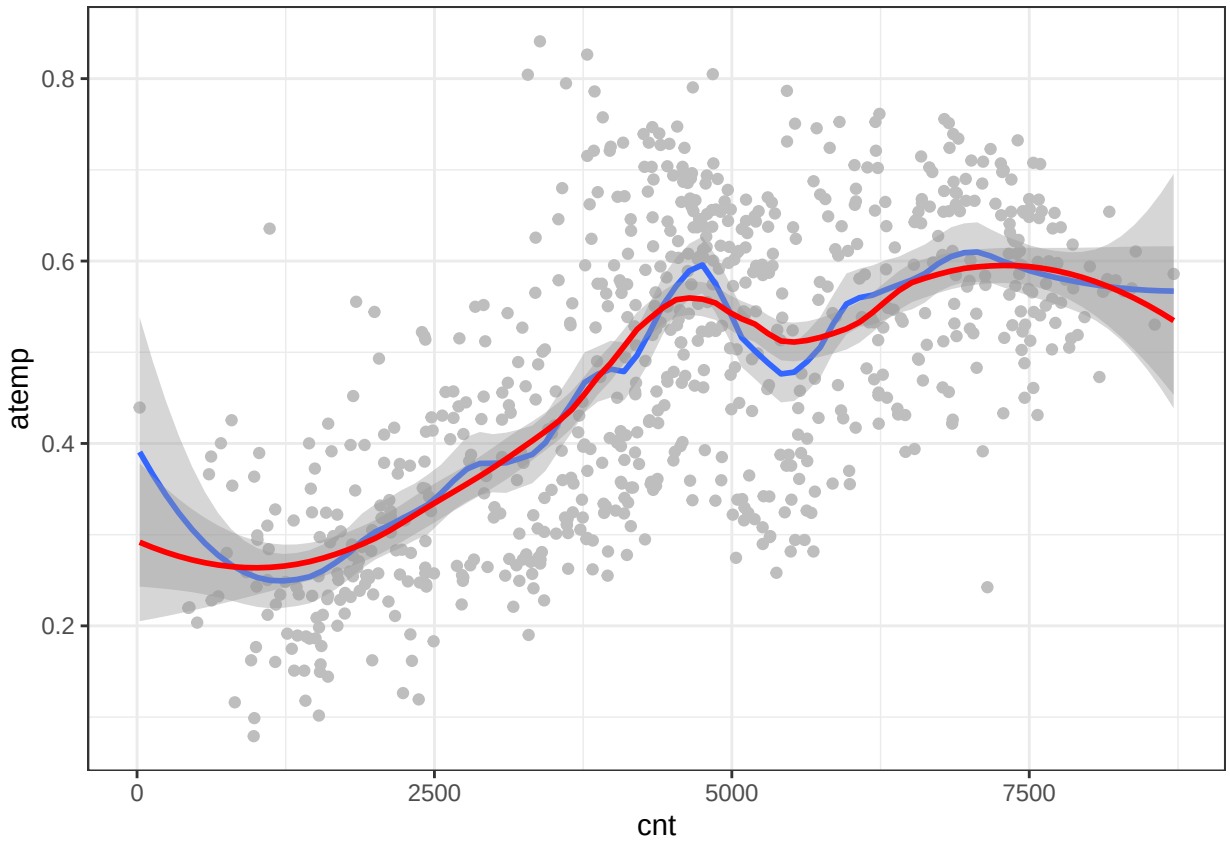
```
## Call:
## smooth.spline(x = bikes$windspeed, y = bikes$cnt, cv = TRUE)
##
## Smoothing Parameter  spar= 1.064815  lambda= 0.009147958 (13 iterations)
## Equivalent Degrees of Freedom (Df): 6.007664
## Penalized Criterion (RSS): 2310995339
## PRESS(l.o.o. CV): 3552035
```

PLOTTING

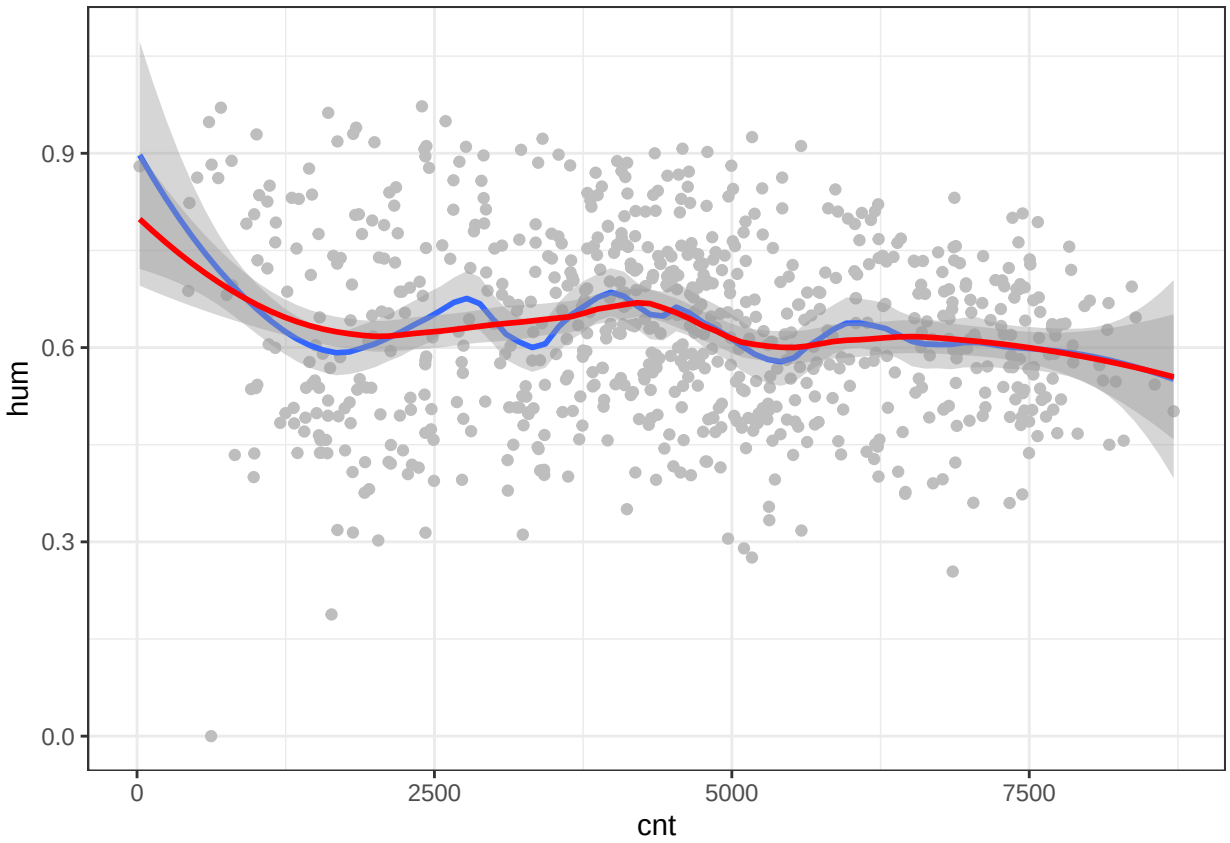
```
ggplot(data = bikes, aes(x = cnt, y = temp)) + geom_point(color = 'gray') +
    geom_smooth(method = 'loess', span = 0.2) +
    geom_smooth(method = 'loess', span = 0.5, color = 'red') +
    theme_bw()
```



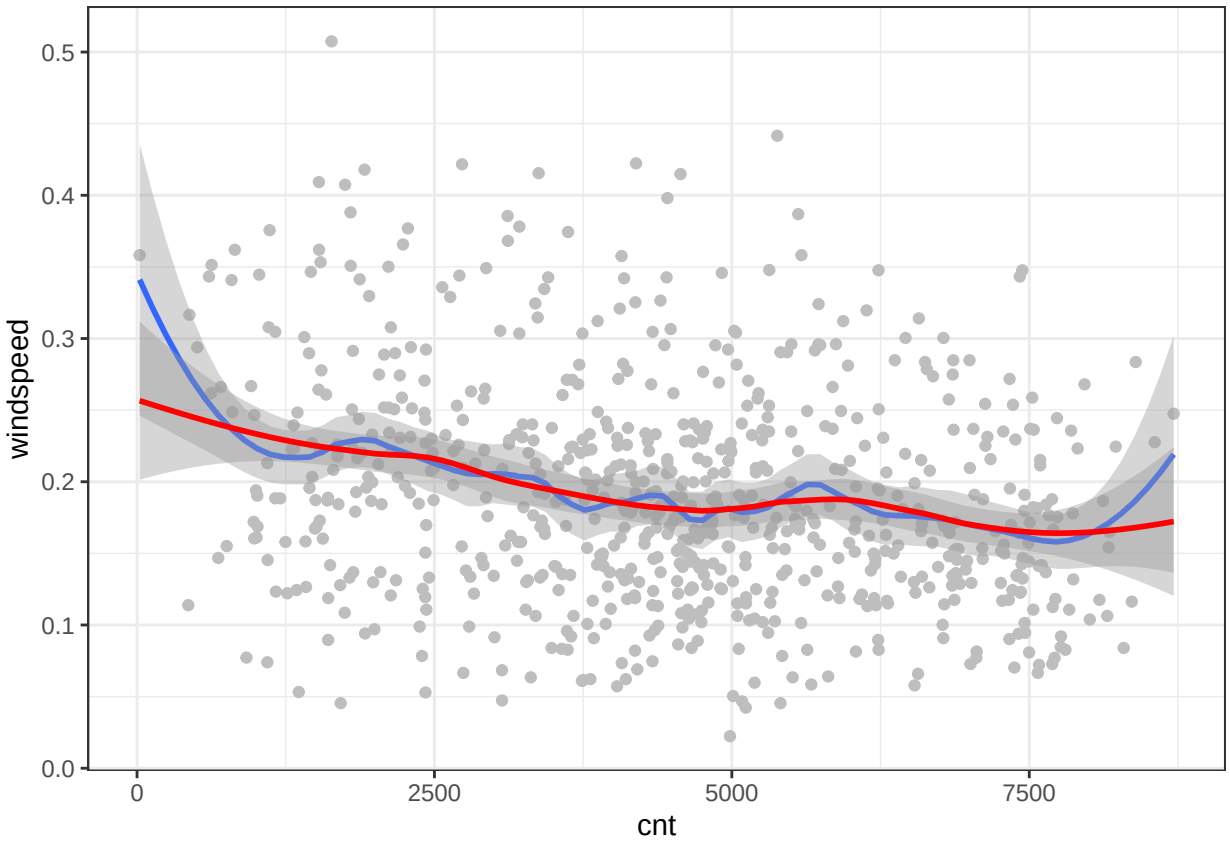

```
ggplot(data = bikes, aes(x = cnt, y = atemp)) + geom_point(color = 'gray') +
    geom_smooth(method = 'loess', span = 0.2) +
    geom_smooth(method = 'loess', span = 0.5, color = 'red') +
    theme_bw()
```




```
ggplot(data = bikes, aes(x = cnt, y = hum)) + geom_point(color = 'gray') +
    geom_smooth(method = 'loess', span = 0.2) +
    geom_smooth(method = 'loess', span = 0.5, color = 'red') +
    theme_bw()
```




```
ggplot(data = bikes, aes(x = cnt, y = windspeed)) + geom_point(color = 'gray') +
    geom_smooth(method = 'loess', span = 0.2) +
    geom_smooth(method = 'loess', span = 0.5, color = 'red') +
    theme_bw()
```



MAKING MODELS

En primer lugar paso a tipo factor las variables categóricas no dummies.

```
bikes$season <- as.factor(bikes$season)
bikes$mnth <- as.factor(bikes$mnth)
bikes$weekday <- as.factor(bikes$weekday)
bikes$weathersit <- as.factor(bikes$weathersit)
```

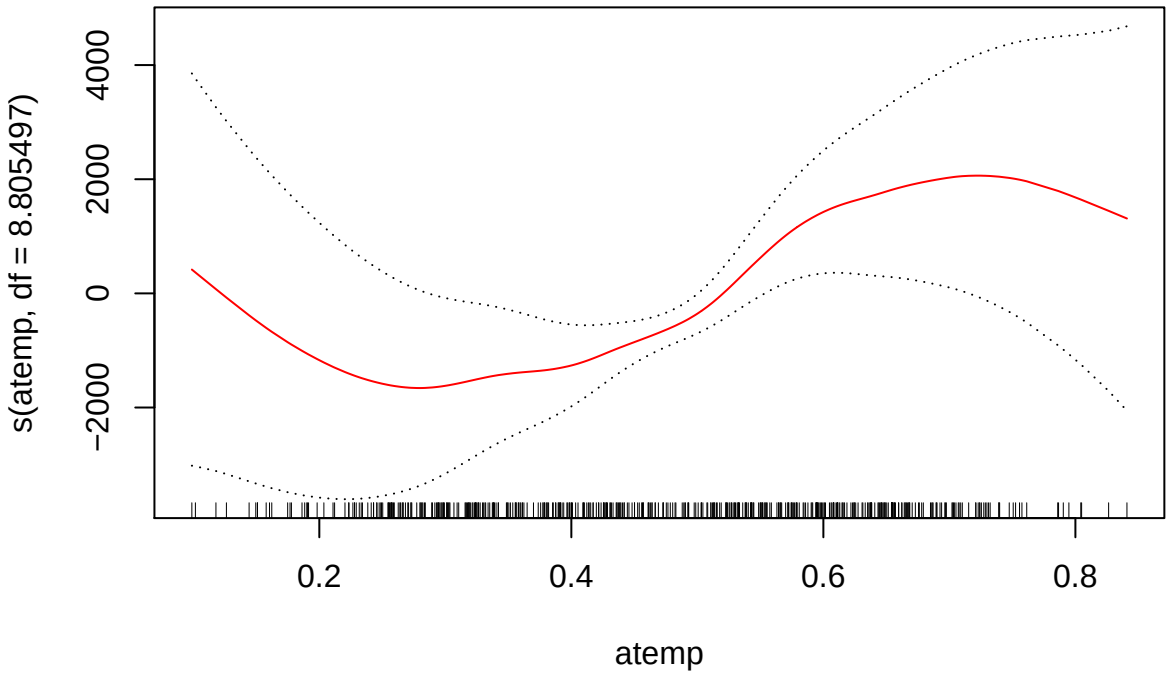
```
train <- sample(nrow(bikes), 0.7*nrow(bikes))

bikes_train <- bikes[train,] # muestra de entrenamiento

bikes_test <- bikes[-train,] # muestra de test
```

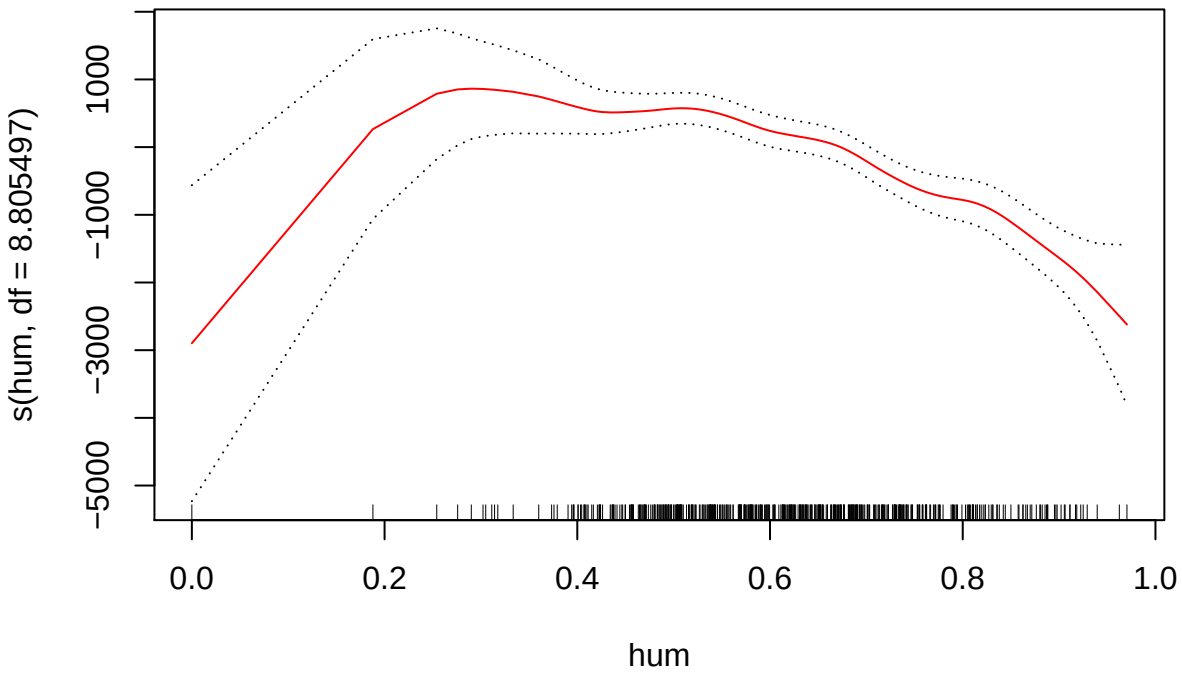
Divido la muestra en train y test. Calculo cada uno de los modelos sobre el train. Realizo la predicción sobre el test, calculo el error medio y aquel modelo con menor error medio es el que pondré a prueba sobre la población total.

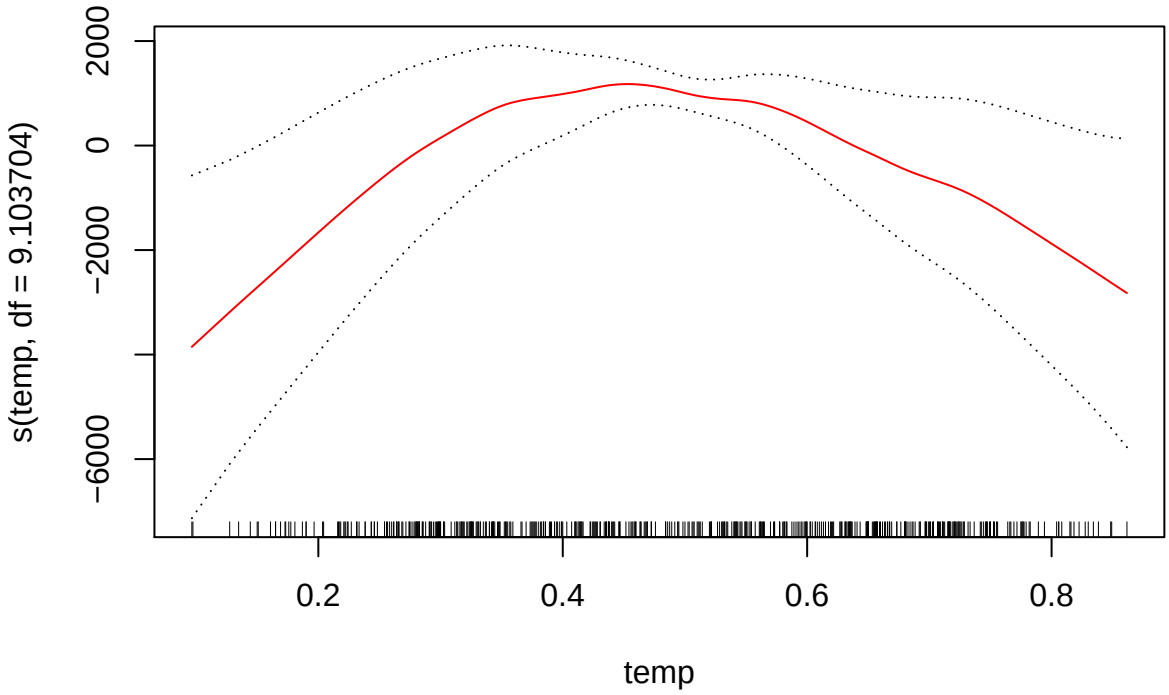
```
# En este modelo incluyo únicamente las variables cuantitativas

modeloGam1 <- gam(cnt ~ s(atemp, df = 8.805497) + s(hum, df = 8.805497) +
                    s(temp, df = 9.103704) + s(windspeed, df = 6.007664), data = bikes_train)
```

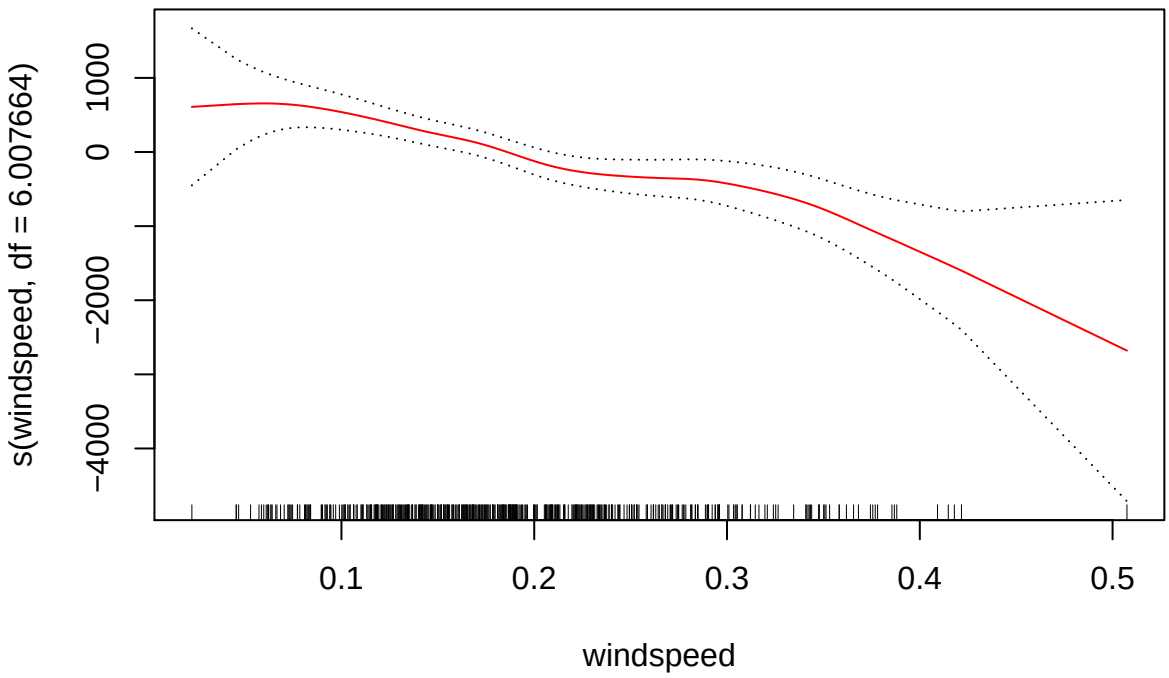
```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

```
plot(modeloGam1, se = TRUE, col = "red")
```




s(hum, df = 8.805497)
1000
−1000
−3000
−5000
0.0
0.2
0.4
0.6
0.8
1.0
hum


s(temp, df = 9.103704)
2000
0
−2000
−6000
0.2
0.4
0.6
0.8
temp



El primer modelo únicamente es elaborado incluyendo las variables cuantitativas de la muestra, excluyo toda variable categórica y/o dummie.

```
summary(modeloGam1)
```

```
##
## Call: gam(formula = cnt ~ s(atemp, df = 8.805497) + s(hum, df = 8.805497) +
##     s(temp, df = 9.103704) + s(windspeed, df = 6.007664), data = bikes_train)
## Deviance Residuals:
##     Min      1Q  Median      3Q     Max
## -2665.3  -928.4  -107.0   962.9  2737.7
##
## (Dispersion Parameter for gaussian family taken to be 1483173)
##
##     Null Deviance: 1836652654 on 510 degrees of freedom
## Residual Deviance: 707886099 on 477.2783 degrees of freedom
## AIC: 8745.866
##
## Number of Local Scoring Iterations: 18
##
## Anova for Parametric Effects
##                                Df    Sum Sq   Mean Sq  F value    Pr(>F)
## s(atemp, df = 8.805497)      1.00 613516778 613516778 413.6516 < 2.2e-16
## s(hum, df = 8.805497)        1.00 100600714 100600714  67.8281 1.726e-15
## s(temp, df = 9.103704)       1.00  10490686  10490686   7.0731  0.008088
## s(windspeed, df = 6.007664)  1.00  74884972  74884972  50.4897 4.386e-12
```

```
## Residuals                   477.28 707886099   1483173
##
## s(atemp, df = 8.805497)     ***
## s(hum, df = 8.805497)       ***
## s(temp, df = 9.103704)      **
## s(windspeed, df = 6.007664) ***
## Residuals
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                             Npar Df Npar F     Pr(F)
## (Intercept)
## s(atemp, df = 8.805497)         7.8 10.845 8.127e-14 ***
## s(hum, df = 8.805497)           7.8  5.213 3.656e-06 ***
## s(temp, df = 9.103704)          8.1 34.604 < 2.2e-16 ***
## s(windspeed, df = 6.007664)     5.0  0.913    0.4726
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

El p-value obtenido para la función del predictor "windspeed" (0.4726) no muestra evidencias de que la relación entre cnt y windspeed no sea lineal, lo que lleva a preguntarse si sería mejor emplear un ajuste lineal en lugar de una smooth spline, reduciendo así la complejidad del modelo.

A continuación procedo a realizar la predicción del modelo sobre la parte test de la muestra.

```
prediccion_modeloGam1 <- predict(modeloGam1, bikes_test)
error_modeloGam1 <- mean((prediccion_modeloGam1 - bikes_test$cnt)^2)
error_modeloGam1
```

```
## [1] 1629602
```

El error medio sobre el test de este primer modelo es de ± 1276,559 bicicletas, es decir, ± 1277 bicicletas aproximadamente.

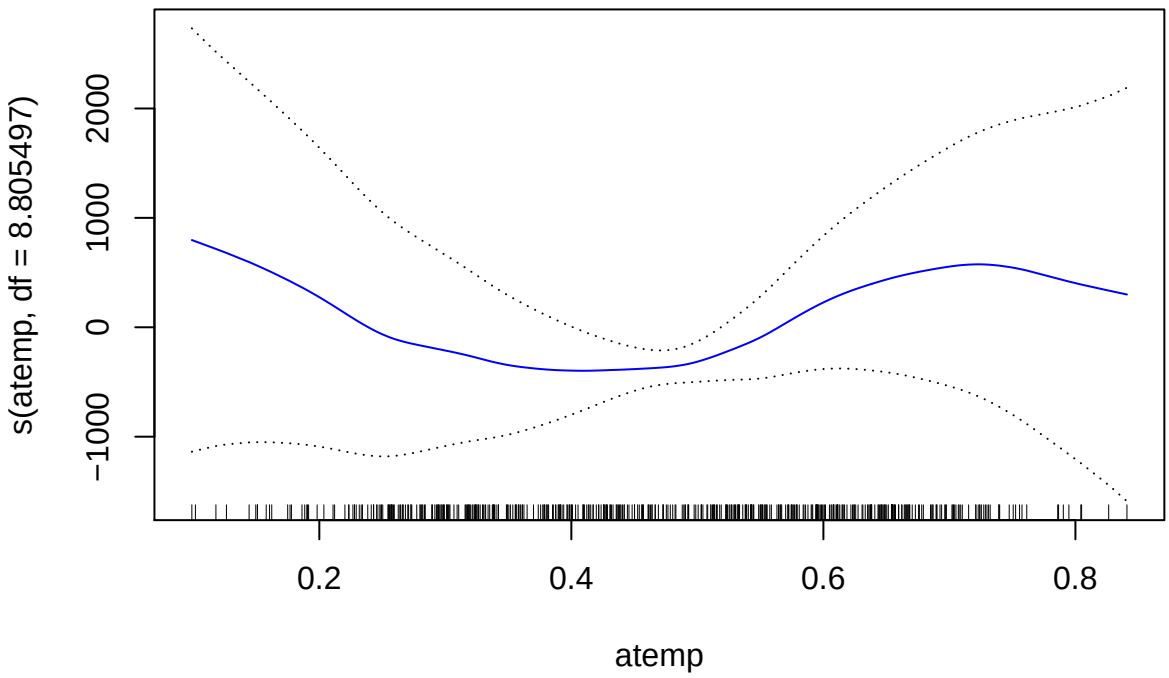
```
sqrt(error_modeloGam1)
```

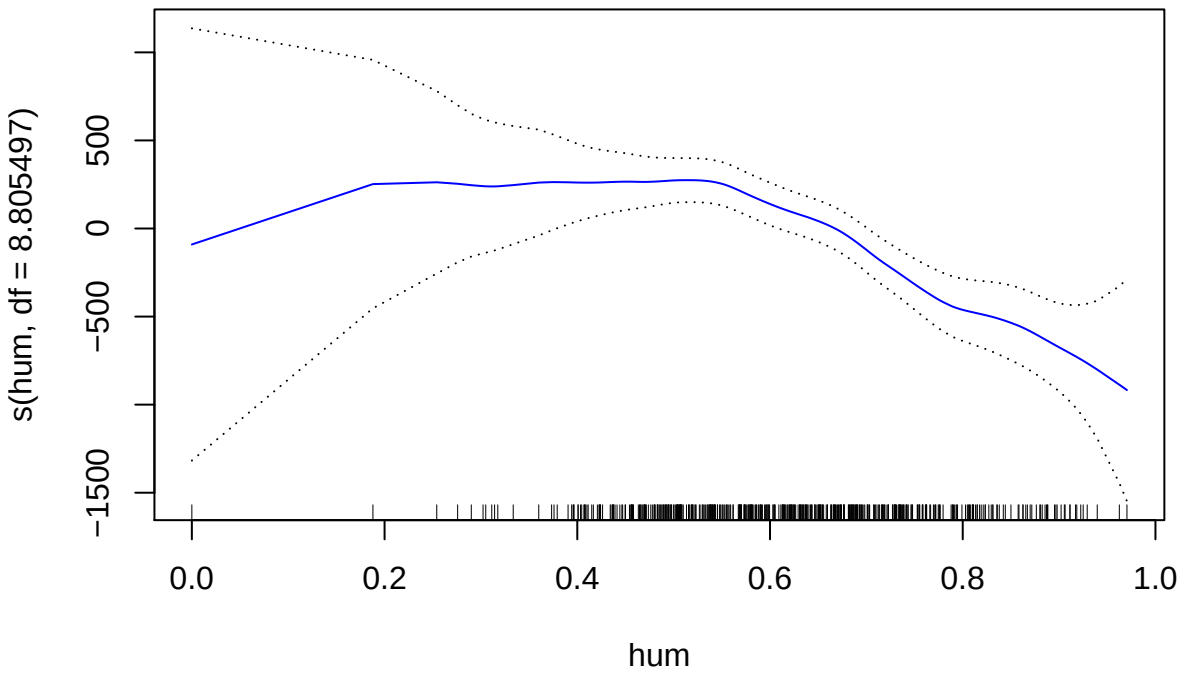
```
## [1] 1276.559
```

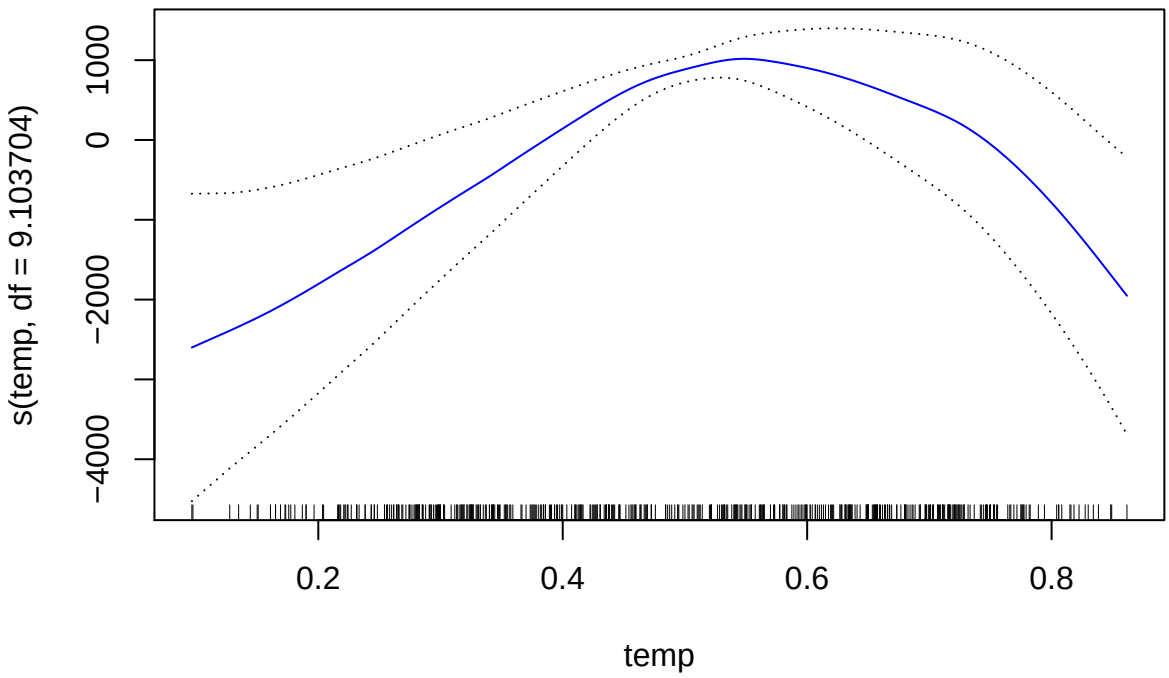
```
# En este modelo incluyo las variables cuantitativas y las categoricas

modeloGam2 <- gam(cnt ~ s(atemp, df = 8.805497) + s(hum, df = 8.805497) + s(temp, df = 9.103704) +
                  s(windspeed, df = 6.007664) + season + yr + mnth + holiday + weekday +
                  workingday + weathersit, data = bikes_train)
```

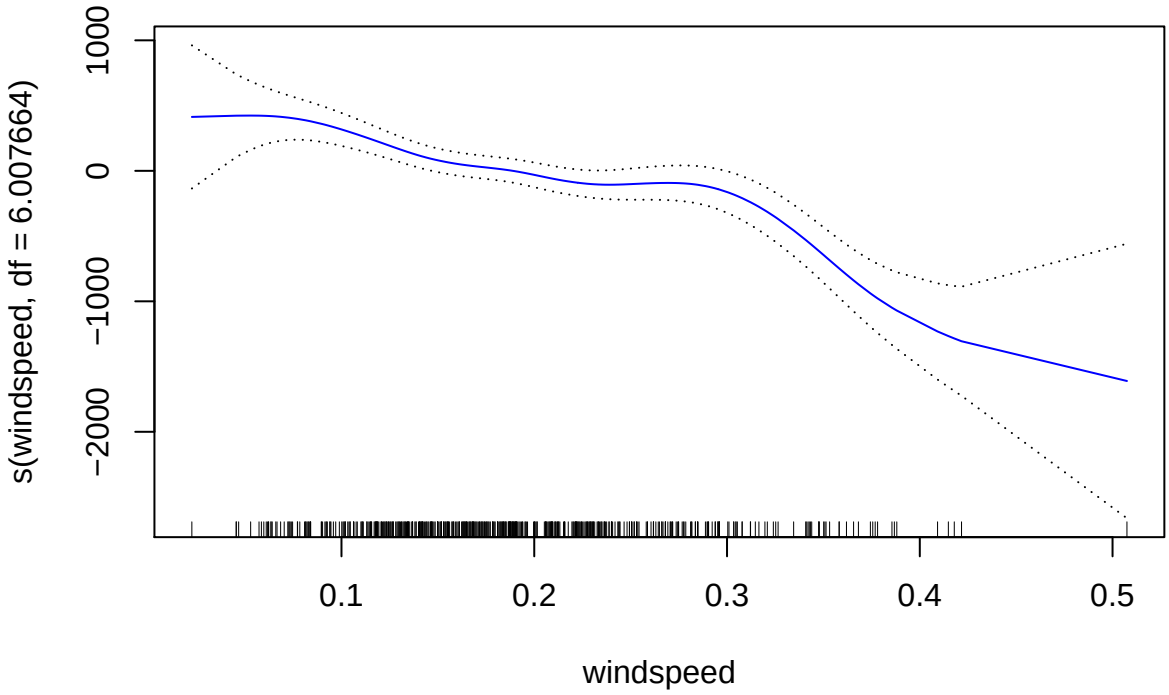
```
## Warning in model.matrix.default(mt, mf, contrasts): non-list contrasts
## argument ignored
```

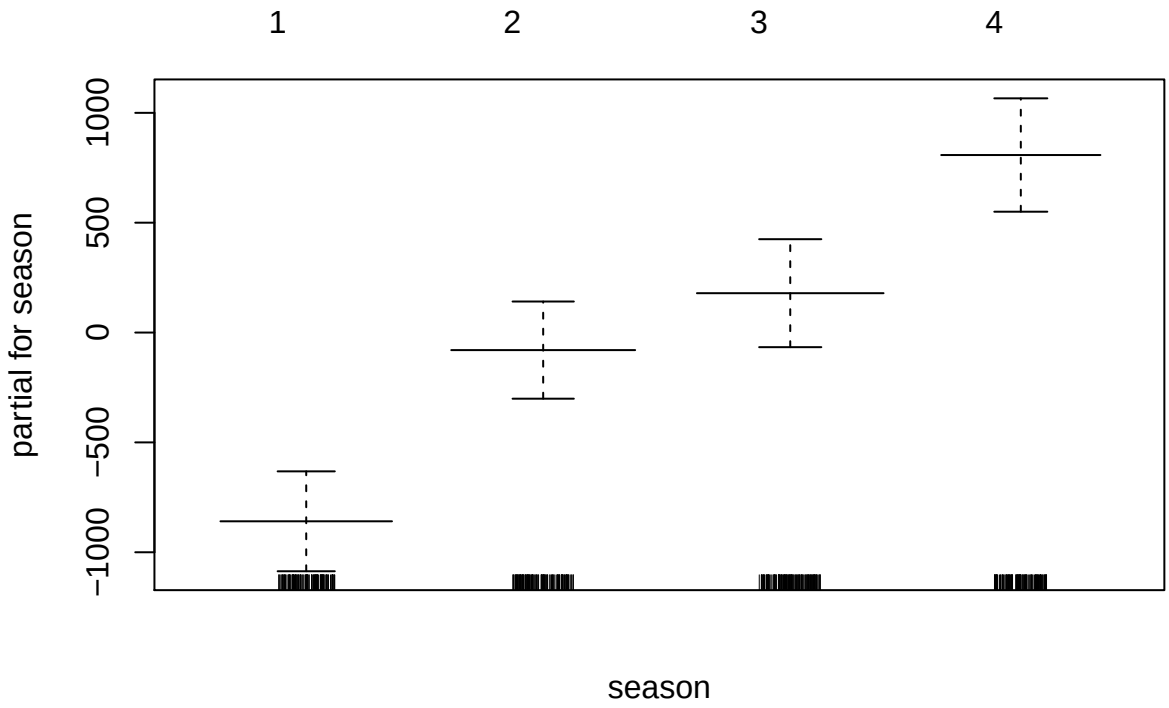
```
plot(modeloGam2, se = TRUE, col = "blue")
```

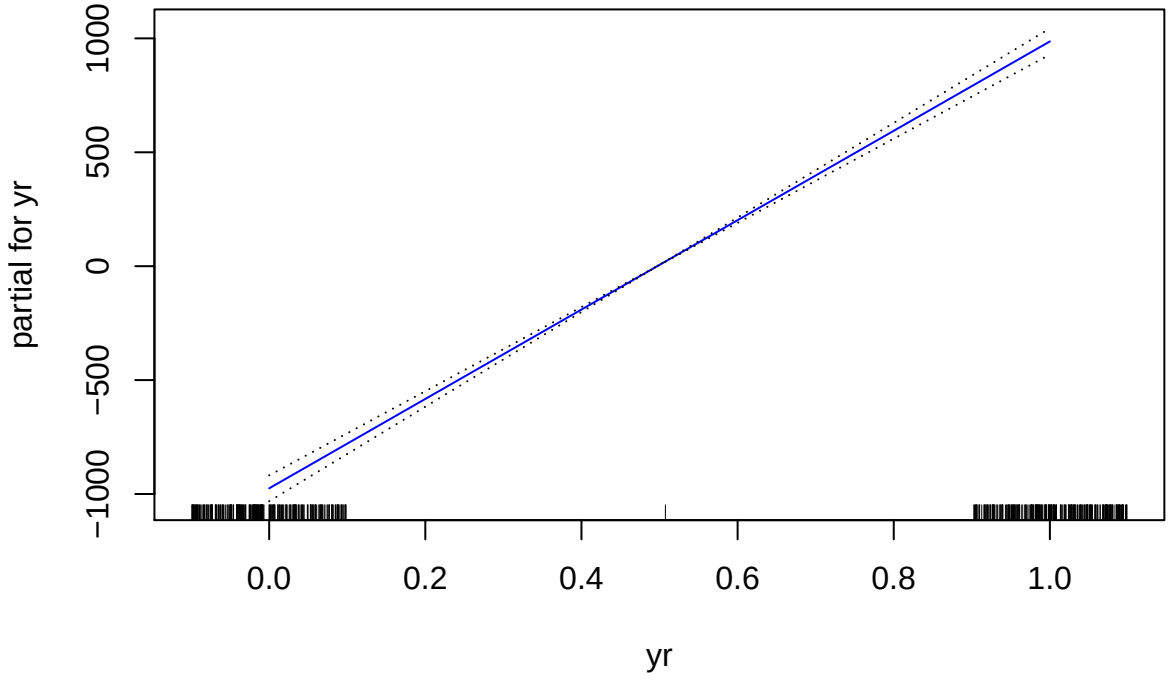

s(atemp, df = 8.805497)
−1000
0
1000
2000
0.2
0.4
0.6
0.8
atemp

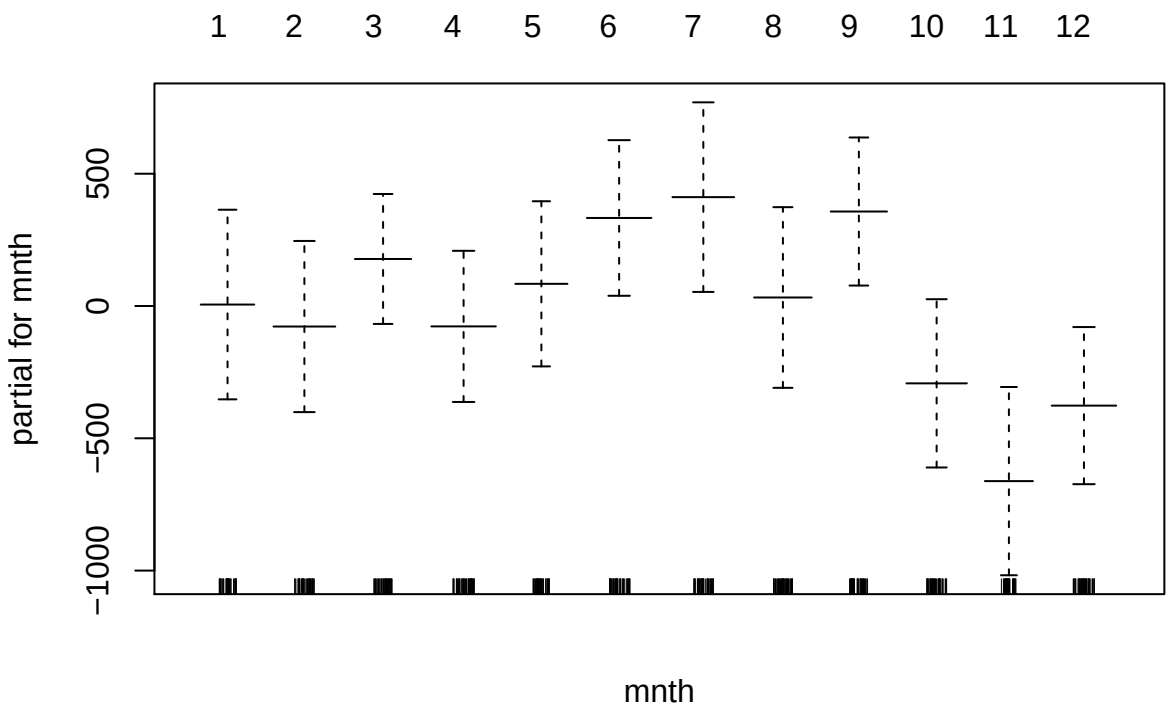

s(hum, df = 8.805497)
−1500
−500
0
500
0.0
0.2
0.4
0.6
0.8
1.0
hum

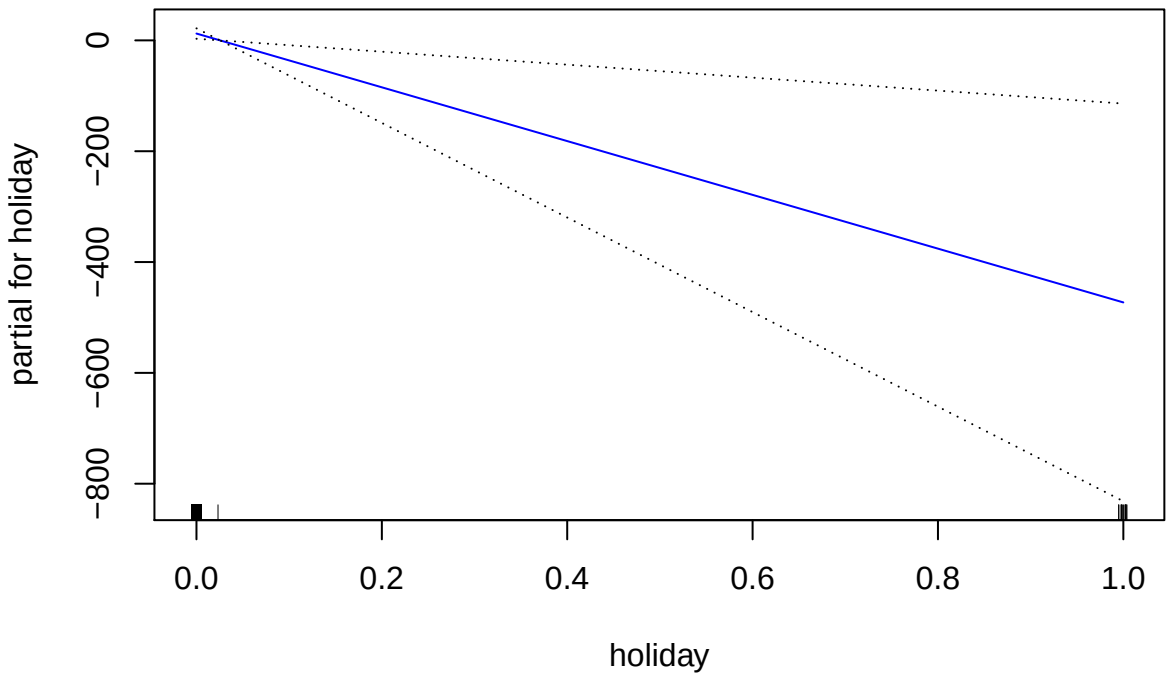

s(temp, df = 9.103704)
1000
0
−2000
−4000
0.2
0.4
0.6
0.8
temp

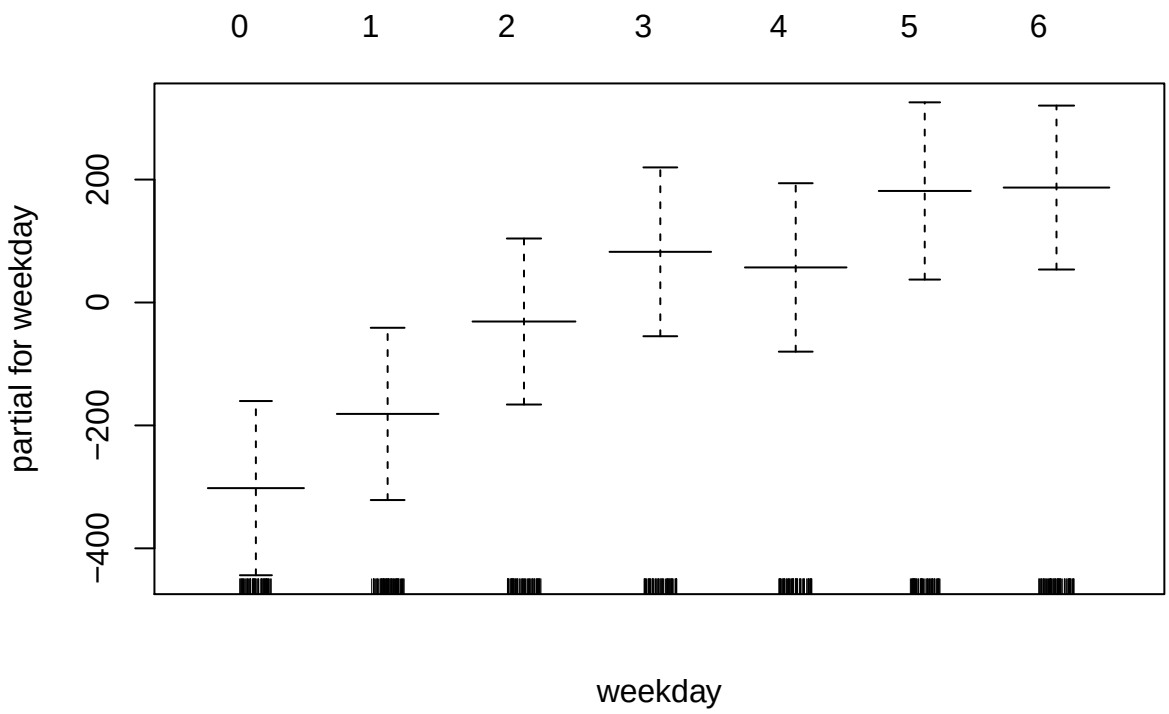

s(windspeed, df = 6.007664)
1000
0
−1000
−2000
0.1
0.2
0.3
0.4
0.5
windspeed

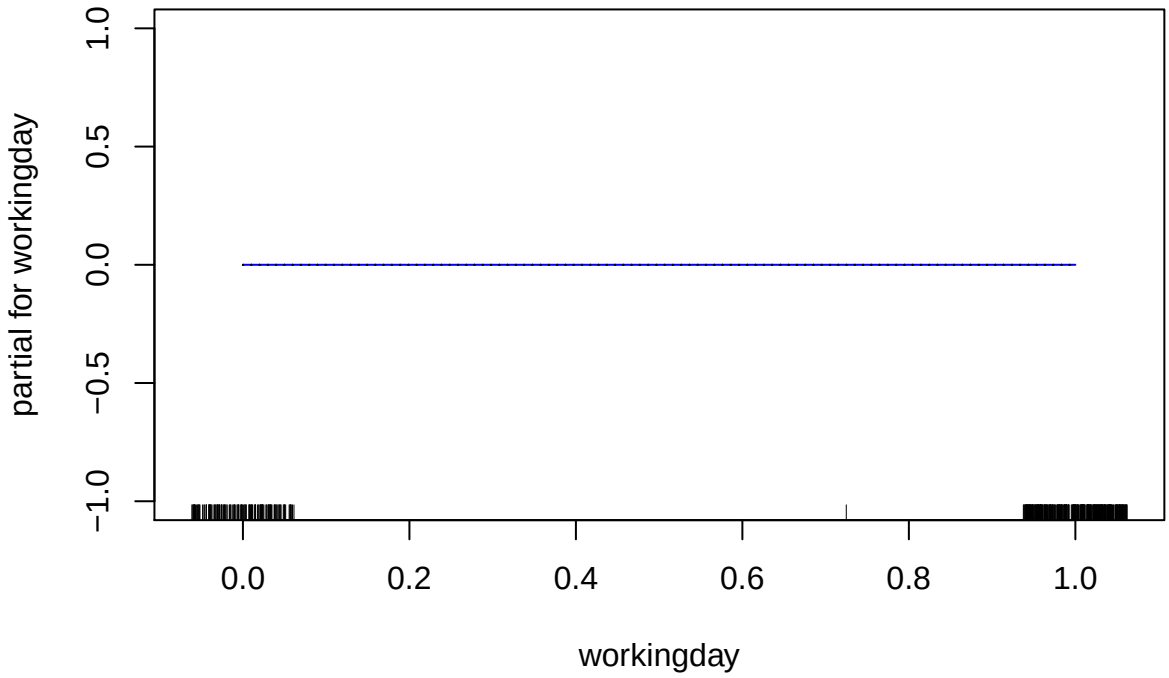

1
2
3
4
partial for season
−1000
−500
0
500
1000
season

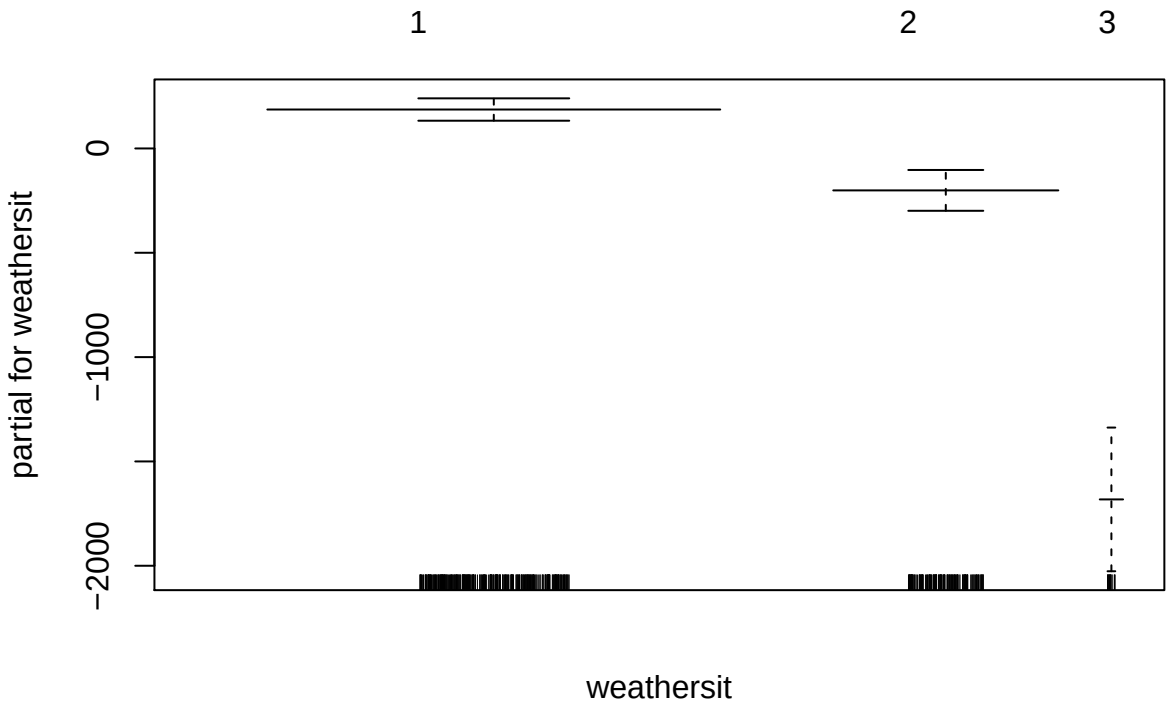

partial for yr
−1000
−500
0
500
1000
0.0
0.2
0.4
0.6
0.8
1.0
yr


1
2
3
4
5
6
7
8
9
10
11
12
partial for mnth
500
0
−500
−1000
mnth


partial for holiday
0
−200
−400
−600
−800
0.0
0.2
0.4
0.6
0.8
1.0
holiday


0
1
2
3
4
5
6
partial for weekday
−400
−200
0
200
weekday


partial for workingday
1.0
0.5
0.0
−0.5
−1.0
0.0
0.2
0.4
0.6
0.8
1.0
workingday



El segundo modelo incorpora todas las variables del DataSet a excepción de "dteday".

```
summary(modeloGam2)
```

```
##
## Call: gam(formula = cnt ~ s(atemp, df = 8.805497) + s(hum, df = 8.805497) +
##     s(temp, df = 9.103704) + s(windspeed, df = 6.007664) + season +
##     yr + mnth + holiday + weekday + workingday + weathersit,
##     data = bikes_train)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2921.45  -295.85    23.05   385.69  1717.74
##
## (Dispersion Parameter for gaussian family taken to be 393771.5)
##
##     Null Deviance: 1836652654 on 510 degrees of freedom
## Residual Deviance: 178488067 on 453.2783 degrees of freedom
## AIC: 8089.83
##
## Number of Local Scoring Iterations: 20
##
## Anova for Parametric Effects
##                                Df    Sum Sq   Mean Sq   F value    Pr(>F)
## s(atemp, df = 8.805497)      1.00 653112956 653112956 1658.6090 < 2.2e-16
## s(hum, df = 8.805497)        1.00  95645757  95645757  242.8966 < 2.2e-16
## s(temp, df = 9.103704)       1.00  12095373  12095373   30.7167 5.074e-08
```

```
## s(windspeed, df = 6.007664)   1.00  72785897  72785897  184.8430 < 2.2e-16
## season                        3.00  69644308  23214769   58.9549 < 2.2e-16
## yr                            1.00 471316716 471316716 1196.9295 < 2.2e-16
## mnth                         11.00  12404826   1127711    2.8639 0.0012042
## holiday                       1.00   3477613   3477613    8.8316 0.0031182
## weekday                       6.00  10804339   1800723    4.5730 0.0001608
## weathersit                    2.00  41246348  20623174   52.3735 < 2.2e-16
## Residuals                   453.28 178488067    393771
##
## s(atemp, df = 8.805497)     ***
## s(hum, df = 8.805497)       ***
## s(temp, df = 9.103704)      ***
## s(windspeed, df = 6.007664) ***
## season                      ***
## yr                          ***
## mnth                        **
## holiday                     **
## weekday                     ***
## weathersit                  ***
## Residuals
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                             Npar Df Npar F     Pr(F)
## (Intercept)
## s(atemp, df = 8.805497)         7.8  5.701 8.295e-07 ***
## s(hum, df = 8.805497)           7.8  2.085 0.0373217 *
## s(temp, df = 9.103704)          8.1 40.271 < 2.2e-16 ***
## s(windspeed, df = 6.007664)     5.0  4.325 0.0007391 ***
## season
## yr
## mnth
## holiday
## weekday
## workingday
## weathersit
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
prediccion_modeloGam2 <- predict(modeloGam2, bikes_test)
error_modeloGam2 <- mean((prediccion_modeloGam2 - bikes_test$cnt)^2)
error_modeloGam2
```

```
## [1] 501094
```

El error medio sobre el test de este segundo modelo es de $\pm$ 707,88 bicicletas, es decir, $\pm$ 708 bicicletas aproximadamente.

```
sqrt(error_modeloGam2)
```

```
## [1] 707.88
```

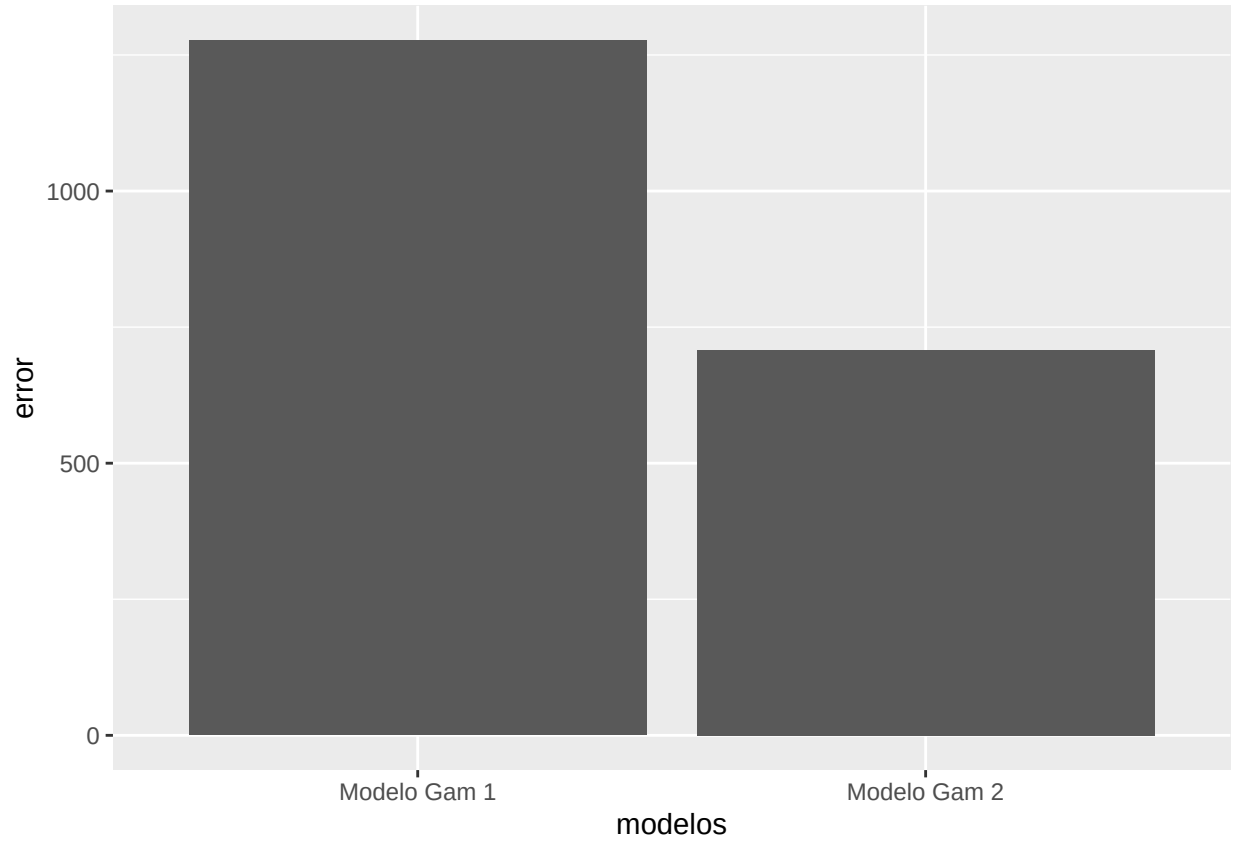
El segundo modelo presenta un error menor. Es por ello que procedo a predecir sobre toda la muestra y calcular el error total.

```
prediccionTotal <- predict(modeloGam2, bikes)
errorTotal <- mean((prediccionTotal - bikes$cnt)^2)
sqrt(errorTotal)
```

```
## [1] 628.4721
```

El error medio sobre la población del modelo finalmente elegigo es de $\pm$ 628,47 bicicletas, lo que significa que se equivoca por término medio en $\pm$ 629 bicicletas.

```
modelos <- c('Modelo Gam 1', 'Modelo Gam 2')
error <- c(sqrt(error_modeloGam1), sqrt(error_modeloGam2))
df <- data.frame(modelos, error)
ggplot(df, aes( x = modelos, y = error)) + geom_col()
```



El segundo modelo presenta un error un 45% menor que el primer modelo en comparación.