Práctica 2: Regresión

Grupo 7 María Gracia Hidalgo Sulbaran, Oscar Mártos Dourado, Guillermo Portela Vázquez

Curso 2024/2025

Esta práctica debe entregarse en formato pdf, incluyendo el código R utilizado, las correspondientes salidas y los comentarios (o interpretaciones de los resultados) pertinentes (para ello se recomienda emplear RMarkdown, a partir de un fichero *.Rmd* o un fichero *.R* mediante spin).

Se empleará el conjunto de datos `amesX` almacenado en el archivo *amesX.RData*, donde X es el número de grupo, con parte de la información que empleó la *Ames Assessor's Office* para calcular los valores de tasación de las propiedades residenciales vendidas en Ames (IA) desde 2006 hasta 2010 (se seleccionaron al azar 1000 propiedades y 9 de las posibles variables explicativas numéricas). Se considerará como respuesta la variable `Sqrt_Price` que contiene la raíz cuadrada del precio de venta y como predictores el resto de variables del conjunto de datos. Para más detalles ver la documentación de `AmesHousing::ames_raw`.

Se debe establecer la semilla igual al número de grupo multiplicado por 10 mediante la función `set.seed()` (también se recomienda hacerlo antes de ajustar cada modelo) y se considerarán el 80% de las observaciones como muestra de aprendizaje y el 20% restante como muestra de test.

Ejercicio 1

Ajustar un modelo lineal con penalización *lasso* empleando la función `glmnet()` del paquete `glmnet`:

```
#Cargamos los datos
load("~/Master_Estadistica/Aprendizaje/practicas/ames7.RData") #cambiar al directorio oportuno

#Cargamos las librerías
library(glmnet)
```

```
## Warning: package 'glmnet' was built under R version 4.3.3
```

```
## Loading required package: Matrix
```

```
## Warning: package 'Matrix' was built under R version 4.3.2
```

```
## Loaded glmnet 4.1-8
```

```
library(mpae)
```

```
## Warning: package 'mpae' was built under R version 4.3.3
```

```
##  mpae: Metodos Predictivos de Aprendizaje Estadistico
##  (Statistical Learning Predictive Methods),
##  version 0.1.2 (built on 2024-03-19).
##  Type `help(mpae)` for an overview of the package or
##  visit https://rubenfcasal.github.io/mpae/.
```

```
#Separamos la muestra en grupos de aprendizaje y test
df <- ames7

set.seed(70)
```

```r
nobs <- nrow(df)
itrain <- sample(nobs, 0.8 * nobs)
train <- df[itrain, ]
test <- df[-itrain, ]

#Como todos los predictores son numéricos, podemos llevar los datos a matriz directamente

x <- as.matrix(subset(train, select = -ncol(train)))

y <- train$Sqrt_Price
```

a. Seleccionar el parámetro λ de regularización por validación cruzada empleando el criterio de un error estándar de Breiman.

   Seleccionamos el valor "óptimo" del hiperparámetro λ (mediante validación cruzada)

```r
set.seed(70)
cv.lasso <- cv.glmnet(x, y, alpha = 1)
```

En este caso el parámetro óptimo, según la regla de un error estándar de Breiman, sería

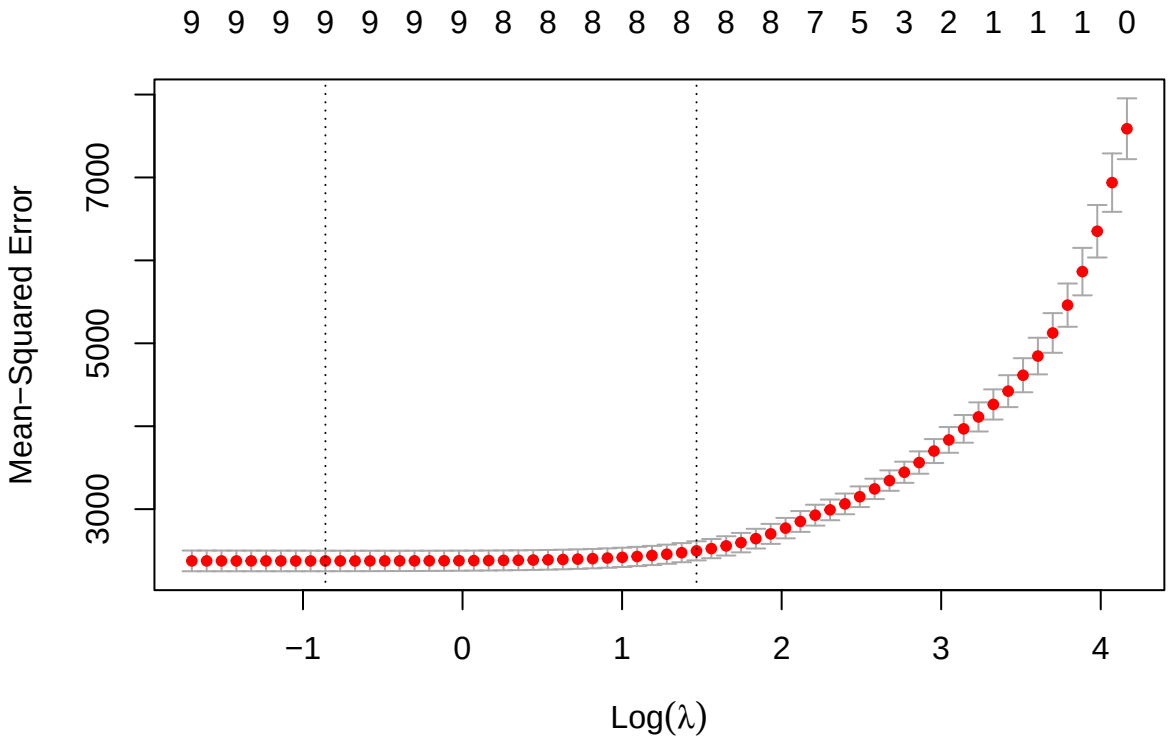
```r
(lambda_breimann <- cv.lasso$lambda.1se)
```

```
## [1] 4.333677
```

que es el valor usado por defecto.

Graficamos la evolución de los errores de validación cruzada en función de la penalización:

```r
plot(cv.lasso)
```



El gráfico muestra la evolución del error. Las líneas verticales son los dos valores de lambda en escala logarítmica: uno es el que minimiza el error global de validación cruzada (el que está más a la izquierda) y el otro es utilizando el criterio de un error estándar de breiman. El modelo más simple, con penalización mayor, es el que usa el error estándar de breiman. Los números en la parte superior del gráfico representan el número de variables con coeficientes distintos de cero.

b. Obtener los coeficientes del modelo y evaluar las predicciones en la muestra de test (gráfico y medidas de error).

Obtenemos los coeficientes del modelo con todas las variables explicativas

```
coef(cv.lasso)
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                               s1
## (Intercept)        289.370073618
## Gr_Liv_Area          0.116595283
## Lot_Area             0.001038292
## Enclosed_Porch      -0.066699893
## Three_season_porch   .
## Bedroom_AbvGr      -11.264195382
## Kitchen_AbvGr      -40.733400267
## Mas_Vnr_Area         0.063483960
## Wood_Deck_SF         0.036033675
## Bsmt_Full_Bath      23.392950299
```

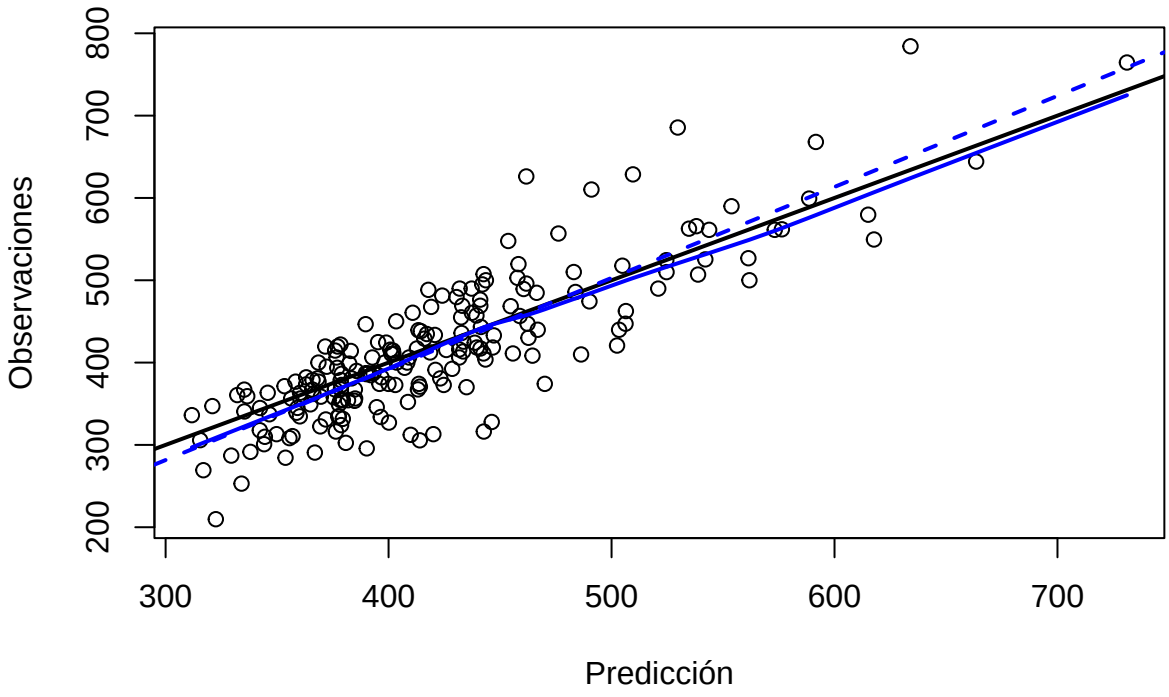
Hay variables con coeficiente positivo y negativo y una sola variable, Three_season_porch, se va a cero. Así que es complicado interpretar este modelo

Evaluamos la precisión en la muestra de test usando el lambda obtenido en el apartado (a)

```
obs <- test$Sqrt_Price
newx <- as.matrix(subset(test, select = -ncol(train)))
pred.lasso <- predict(cv.lasso, newx = newx)
```

Comparamos los valores observados en la muestra de entrenamiento con los resultados del modelo.

```
pred.plot(pred.lasso, obs, xlab = "Predicción", ylab = "Observaciones")
```



Se aprecia que en unas pocas predicciones a partir de 470 se quedan unusualmente por encima de la recta $x = y$ pero en general el comportamiento es bastante bueno

```
(acc.lasso <-accuracy(pred.lasso, obs))
```

```
##          me       rmse        mae        mpe       mape  r.squared
## -5.4934960 46.9381382 35.4566449 -2.8847650  8.8971130  0.7227966
```

Con este modelo explicaríamos un 72% de la variabilidad del Sqrt_Price en nuevas observaciones.

De los errores podemos interpretar:

- RMSE: Es una medida de la magnitud promedio del error (es decir, las diferencias entre los valores observados y los predichos). En este caso, un RMSE de 46.94 sugiere que las predicciones se desvían de los valores observados en promedio por alrededor de 47 unidades.
- MAE: Es el promedio de los valores absolutos de los errores. Aquí, el modelo tiene un error promedio absoluto de 35.46 unidades, lo que indica una desviación media moderada en las predicciones.
- MPE: Calcula el error promedio en porcentaje, indicando si las predicciones están por encima o por debajo de los valores observados en términos relativos. Un valor negativo indica subestimación, por lo

que el modelo tiende a predecir un 2.88% menos que los valores reales, en promedio.

- MAPE: Es el error porcentual absoluto promedio. En este caso, el error absoluto promedio es del 8.89%, lo cual puede ser considerado como una precisión aceptable dependiendo del contexto y del rango de los valores observados.
- R-SQUARED: Mide la proporción de la varianza en los valores observados que es explicada por el modelo. Un R-SQUARED de 0.72 significa que el modelo explica aproximadamente el 72% de la variación en los datos observados, lo que sugiere que el modelo captura una gran parte de la relación entre las variables predictoras y el resultado, pero no todas.

c. ¿Cuál sería el número de coeficientes distintos de cero si se selecciona λ de forma que minimice el error de validación cruzada?

```
coef(cv.lasso, s = "lambda.min")
```

```
## 10 x 1 sparse Matrix of class "dgCMatrix"
##                                 s1
## (Intercept)         301.799465459
## Gr_Liv_Area           0.130134853
## Lot_Area              0.001679866
## Enclosed_Porch       -0.119223907
## Three_season_porch    0.027932536
## Bedroom_AbvGr       -20.705423100
## Kitchen_AbvGr       -53.702385203
## Mas_Vnr_Area          0.066031756
## Wood_Deck_SF          0.045363376
## Bsmt_Full_Bath       27.442657445
```

Esta vez ningún coeficiente es exactamente igual a cero

Ejercicio 2

Ajustar un modelo mediante regresión spline adaptativa multivariante (MARS) empleando el método `"earth"` del paquete `caret`:

a. Utilizar validación cruzada con 5 grupos para seleccionar los valores "óptimos" de los hiperparámetros considerando las posibles combinaciones de `degree = 1:2` y `nprune = c(5, 10, 15)` y empleando el criterio de un error estándar de Breiman.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.3
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
## Loading required package: lattice
```

```
#Para la selección de los hiperparámetros óptimos, consideramos una rejilla de búsqueda personalizada
tuneGrid <- expand.grid(degree = 1:2, nprune = c(5, 10, 15))
train_control <- trainControl(method = "cv", number = 5, selectionFunction = "oneSE")
```

Calculamos los errores para cada combinación de valores de los hiperparámetros.

```
set.seed(70)
caret.mars <- train(Sqrt_Price ~ ., data = ames7, method = "earth",
                    trControl = train_control, tuneGrid = tuneGrid)
```

```
## Loading required package: earth
```

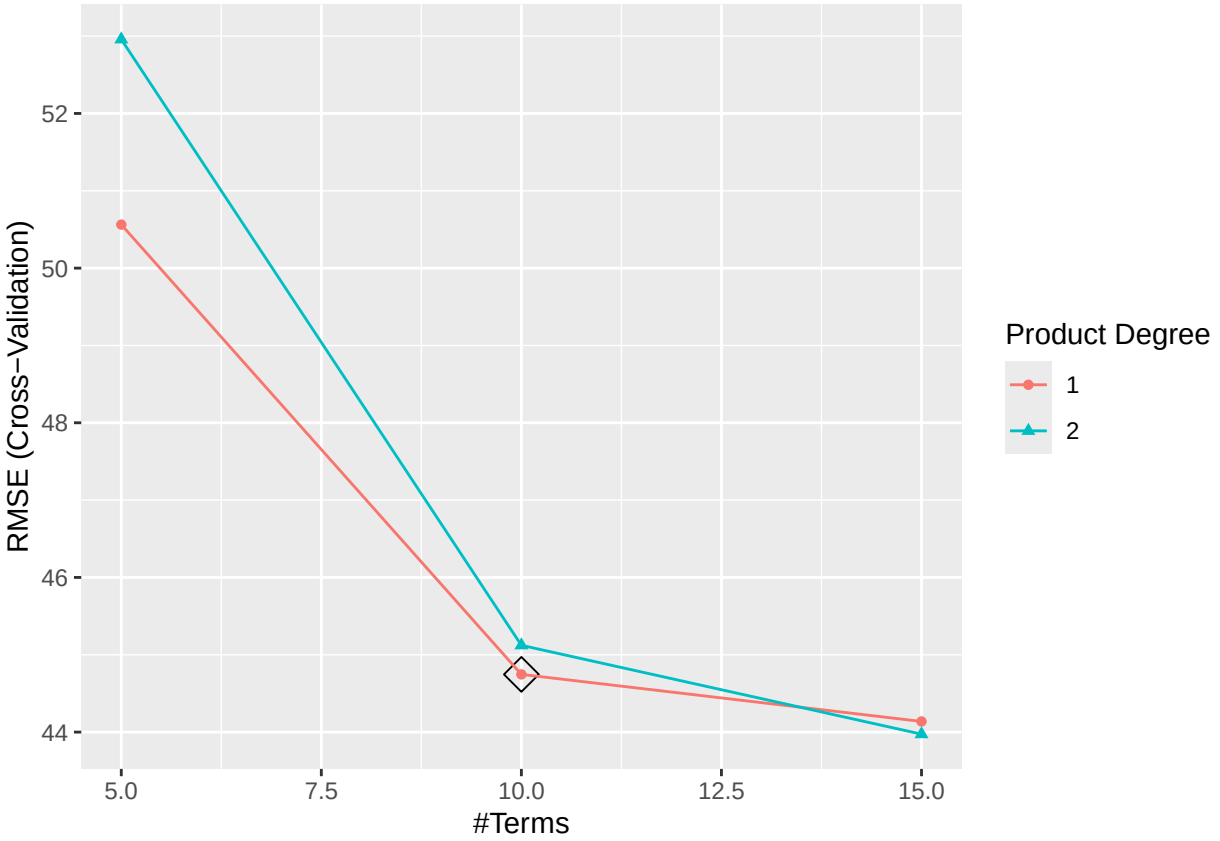
```
## Warning: package 'earth' was built under R version 4.3.3
## Loading required package: Formula
## Loading required package: plotmo
## Warning: package 'plotmo' was built under R version 4.3.3
## Loading required package: plotrix
## Warning: package 'plotrix' was built under R version 4.3.2
```

```
caret.mars
```

```
## Multivariate Adaptive Regression Spline
##
## 1000 samples
##    9 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 800, 800, 800, 800, 800
## Resampling results across tuning parameters:
##
##   degree  nprune  RMSE      Rsquared   MAE
##   1        5      50.56257  0.6700685  37.75728
##   1       10      44.74624  0.7407289  33.87256
##   1       15      44.13752  0.7477412  33.32871
##   2        5      52.95844  0.6377047  40.04453
##   2       10      45.12251  0.7357216  33.87640
##   2       15      43.97328  0.7500880  33.19475
##
## RMSE was used to select the optimal model using  the one SE rule.
## The final values used for the model were nprune = 10 and degree = 1.
```

Los valores óptimos se obtienen con 10 términos en el modelo y con iteración de orden 1. RMSE = 44.74624, Rsquared = 0.7407289, MAE = 33.87256

```
ggplot(caret.mars, highlight = TRUE)
```



La representación gráfica nos muestra la combinación de hiperparámetros, siendo el óptimo el punto que está encerrado en el rombo.

```
summary(caret.mars$finalModel)
```

```
## Call: earth(x=matrix[1000,9], y=c(305.6,461.5,5...), keepxy=TRUE, degree=1,
##             nprune=10)
##
##                       coefficients
## (Intercept)              359.97036
## h(984-Gr_Liv_Area)        -0.24349
## h(Gr_Liv_Area-984)         0.12606
## h(13891-Lot_Area)         -0.00468
## h(34-Enclosed_Porch)       1.09982
## h(3-Bedroom_AbvGr)        24.64761
## h(Bedroom_AbvGr-3)       -30.47140
## h(2-Kitchen_AbvGr)        47.70525
## h(738-Mas_Vnr_Area)       -0.07515
## h(1-Bsmt_Full_Bath)      -26.62187
##
## Selected 10 of 19 terms, and 7 of 9 predictors (nprune=10)
## Termination condition: Reached nk 21
## Importance: Gr_Liv_Area, Bedroom_AbvGr, Bsmt_Full_Bath, Enclosed_Porch, ...
## Number of terms at each degree of interaction: 1 9 (additive model)
## GCV 2024.473    RSS 1948274    GRSq 0.7368187    RSq 0.7462173
```

El modelo final contiene 15 términos con interacciones. El crecimiento terminó cuando se alcanzaron 21

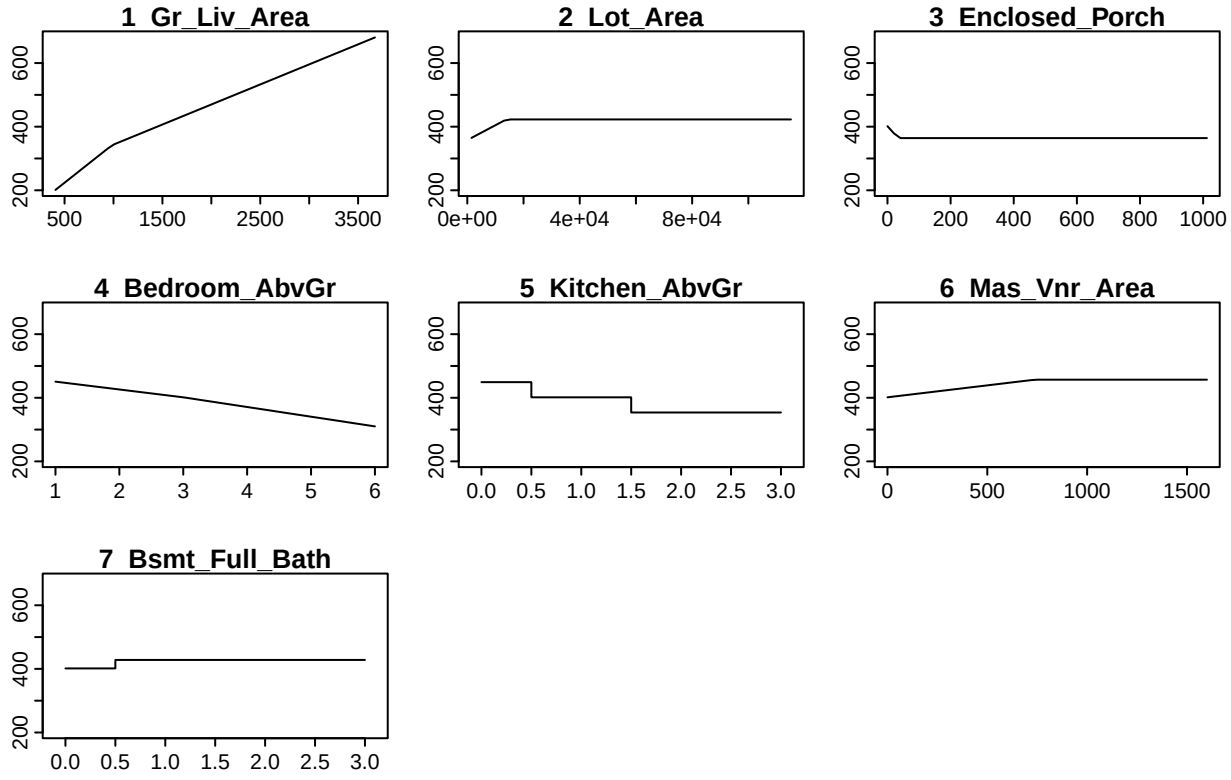
componentes. Tenemos 7 predictores incluidos en el modelo, de los 9 que había. Tenemos una constante y 9 efectos principales (no hay interacciones). Una estimación de la variabilidad explicada directamente a partir de la muestra de entrenamiento es GRSq = 0.7368187.

b. Estudiar el efecto de los predictores incluidos en el modelo final y obtener medidas de su importancia.

Representamos los efectos parciales de las componentes:

```
plotmo(caret.mars$finalModel, caption = "")
```

```
##  plotmo grid:    Gr_Liv_Area Lot_Area Enclosed_Porch Three_season_porch
##                       1458.5   9333.5              0                   0
##  Bedroom_AbvGr Kitchen_AbvGr Mas_Vnr_Area Wood_Deck_SF Bsmt_Full_Bath
##              3             1            0            0              0
```



- Gr_Liv_Area: Tiene un efecto positivo. Cambia de gradiente en 984
- Lot_Area: Inicialmente tiene un efecto positivo que luego el efecto se mantiene nulo. Cambia de gradiente en 13891
- Enclosed_Porch: Inicialmente tiene un efecto negativo que luego se mantiene nulo. Cambia de gradiente en 34.
- Bedroom_AbvGr: Tiene efecto negativo. Cambia ligeramente de gradiente en 3.
- Kitchen_AbvGr: Tiene efecto nulo en varios tramos. Cambia de gradiente en 2.
- Mas_Vnr_Area: Tiene efecto positivo. Cambia de gradiente en 738.
- Bsmt_Full_Bath: Tiene efecto nulo en varios tramos. Cambia de gradiente en 1.

Obtenemos medidas de la importancia de las variables

```r
varimp <- evimp(caret.mars$finalModel)

varimp
```

```
##                nsubsets   gcv    rss
## Gr_Liv_Area           9 100.0  100.0
## Bedroom_AbvGr         8  53.2   53.8
## Bsmt_Full_Bath        7  43.4   44.1
## Enclosed_Porch        6  35.9   36.6
## Mas_Vnr_Area          5  29.8   30.5
## Lot_Area              4  25.2   25.8
## Kitchen_AbvGr         1  11.1   11.4
```

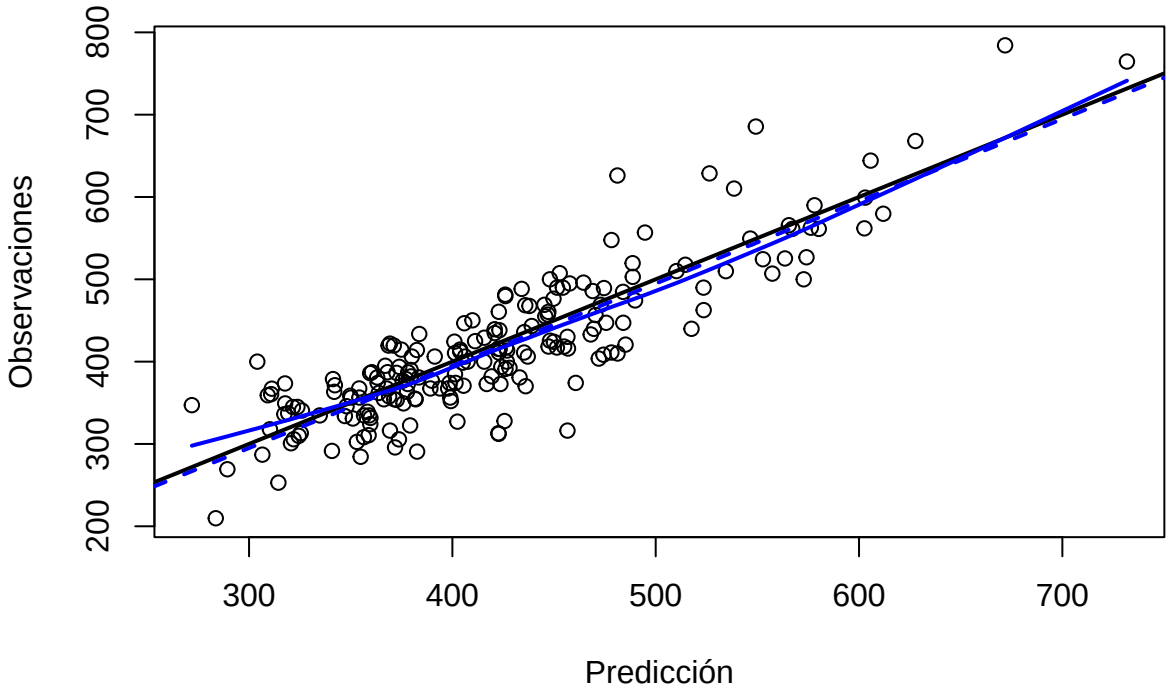
Gr_Liv_Area aparece en 9 subconjuntos, lo que sugiere que es la variable más utilizada en las diferentes interacciones del modelo. Adicionalmente, tiene el mayor GCV (100), lo que indica que es la variable más importante para la predicción según este criterio. Las demás variables tienen valores menores, lo que significa que aportan menos al modelo. Un rss más alto indica que la variable está contribuyendo más a reducir el error residual.

Gr_Liv_Area es la variable más importante para el modelo, contribuyendo significativamente a la predicción y al ajuste del modelo, seguida por Bedroom_AbvGr y Bsmt_Full_Bath. Las variables como Enclosed_Porch, Mas_Vnr_Area, y Lot_Area también aportan al modelo, pero en menor medida. Kitchen_AbvGr es la variable menos importante, con una contribución mucho menor según estas métricas.

c. Evaluar las predicciones en la muestra de test.

```r
pred.mars <- predict(caret.mars, newdata = test)

#Gráfico
pred.plot(pred.mars, obs, xlab = "Predicción", ylab = "Observaciones")
```



Los valores no parecen alejarse mucho de la recta $x = y$, lo que tal vez más resalta son unas predicciones entre 470 y 550 que resultaron demasiado grandes, esto era algo que ya se observaba en el anterior pred.plot, una cosa que difiere del anterior pred.plot es que la recta de regresión acaba por debajo de la recta $x = y$

```
#Medidas de Error
(acc_2 <-accuracy(pred.mars, test$Sqrt_Price))
```

```
##         me       rmse        mae        mpe       mape  r.squared
## -5.0563718 43.4000250 33.5690006 -2.2949402  8.4387905  0.7630118
```

Con este modelo explicaríamos un 76% de la variabilidad del Sqrt_Price en nuevas observaciones.

De los errores podemos interpretar:

- RMSE: Es una medida de la magnitud promedio del error (es decir, las diferencias entre los valores observados y los predichos). En este caso, un RMSE de 43.4 sugiere que las predicciones se desvían de los valores observados en promedio por alrededor de 43 unidades.
- MAE: Es el promedio de los valores absolutos de los errores. Aquí, el modelo tiene un error promedio absoluto de 33.56 unidades, lo que indica una desviación media moderada en las predicciones.
- MPE: Calcula el error promedio en porcentaje, indicando si las predicciones están por encima o por debajo de los valores observados en términos relativos. Un valor negativo indica subestimación, por lo que el modelo tiende a predecir un 2.29% menos que los valores reales, en promedio.
- MAPE: Es el error porcentual absoluto promedio. En este caso, el error absoluto promedio es del 8.43%, lo cual puede ser considerado como una precisión aceptable dependiendo del contexto y del rango de los valores observados.
- R-SQUARED: Mide la proporción de la varianza en los valores observados que es explicada por el modelo. Un R-SQUARED de 0.76 significa que el modelo explica aproximadamente el 76% de la

variación en los datos observados, lo que sugiere que el modelo captura una gran parte de la relación entre las variables predictoras y el resultado, pero no todas.

Ejercicio 3

Ajustar un modelo mediante regresión por projection pursuit empleando la función `ppr()`:

a. Considerar una única función ridge y seleccionar el suavizado máximo `bass = 10`.

En primer lugar ajustamos un modelo PPR con un solo término

```
set.seed(70)
ppreg <- ppr(ames7$Sqrt_Price ~ ., nterms = 1, data = ames7, bass = 10)
```

b. Obtener los coeficientes del modelo y representar la función ridge (comentar).

```
summary(ppreg)
```

```
## Call:
## ppr(formula = ames7$Sqrt_Price ~ ., data = ames7, nterms = 1,
##     bass = 10)
##
## Goodness of fit:
## 1 terms
## 2220862
##
## Projection direction vectors ('alpha'):
##        Gr_Liv_Area           Lot_Area     Enclosed_Porch Three_season_porch
##       2.493617e-03       2.401821e-05      -2.568590e-03       5.554603e-04
##      Bedroom_AbvGr      Kitchen_AbvGr       Mas_Vnr_Area       Wood_Deck_SF
##      -4.164460e-01      -7.617285e-01       1.296443e-03       9.295629e-04
##     Bsmt_Full_Bath
##       4.963132e-01
##
## Coefficients of ridge terms ('beta'):
##   term 1
## 73.86525
```

```
cat("La media de Sqrt_Price es", ppreg$yb, "\n")
```

```
## La media de Sqrt_Price es 417.5001
```

Comentario Guille Como estamos considerando únicamente una función ridge, el modelo recién ajustado se trata de un modelo single-index. Por tanto, la expresión genérica (suponiendo que las funciones ridge están reescaladas)

$$m_i(\mathbf{x}) = \beta_{i0} + \sum_{m=1}^{M} \beta_{im} g_m(\alpha_{1m} x_1 + \alpha_{2m} x_2 + \ldots + \alpha_{pm} x_p)$$

expresada con los valores obtenidos sería la siguiente:

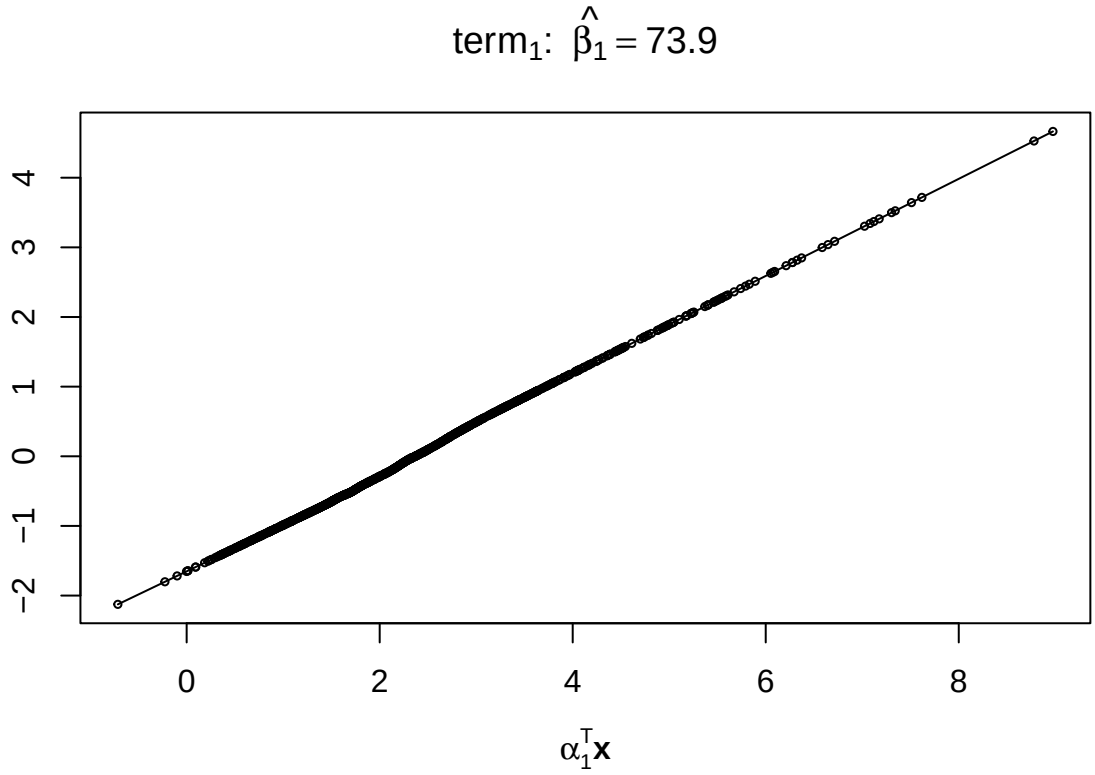
$$m(\mathbf{x}) = 417.5 + 73.8625 g(2.493617e-03\ x_1 + 2.401821e-05\ x_2 + \ldots + 4.963132e-01\ x_p)$$

Donde g es la función de supersuavizado de Friedman Gustaría tener una descripción algo más detallada de lo que significa cada variable en la documentación aportada. Por ejemplo Kitchen_abvGr,la que tiene el coeficiente de mayor valor absoluto no aparece siquiera en la documentación. Esto juntado a no conocer muy bien como funciona la función g dificulta dar una interpretación de los parámetros del modelo... si hacemos un pequeño análisis exploratorio de los datos podemos observar que las variables - Bedroom_AbvGr - Kitchen_AbvGr - Bsmt_Full_Bath

Son las únicas categóricas, las que tienen los valores numéricos más bajos, esto seguramente sea lo que explica que sean las variables a las cuales el modelo les dá las ponderaciones más grandes (en valor absoluto).

Podemos representar las funciones rigde con

```
plot(ppreg)
```

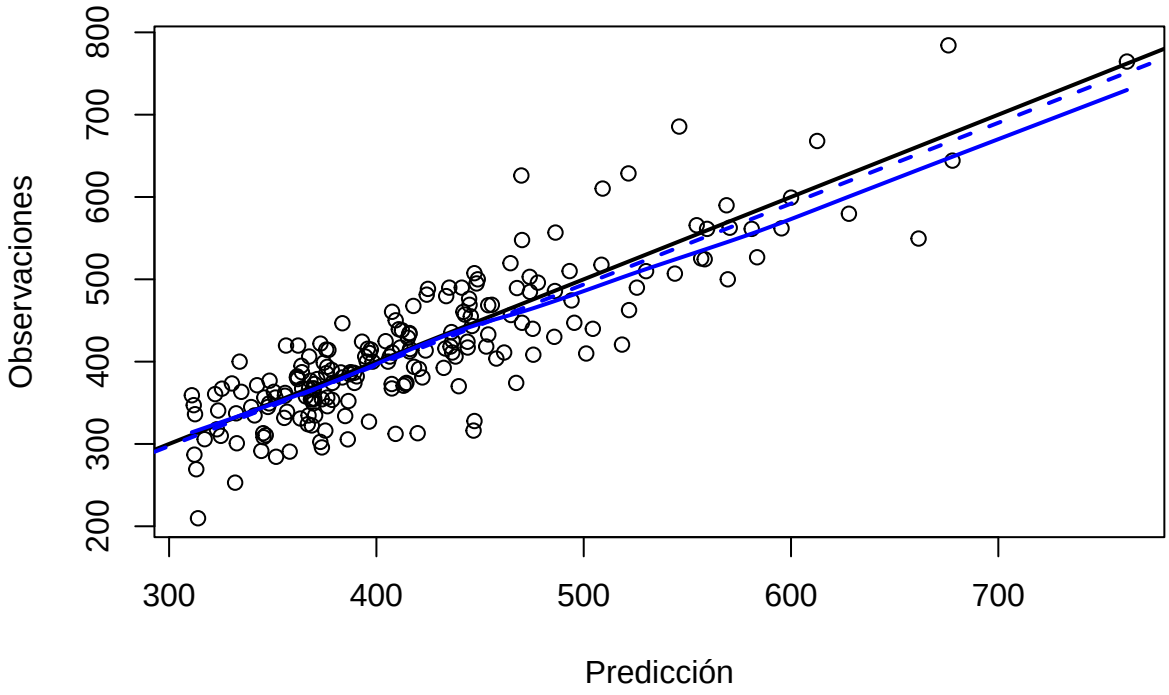


Se aprecia una linea recta lo cual hace que la interpretación no pueda ser muy sofisticada.

c. Evaluar las predicciones en la muestra de test (gráfico y medidas de error) y comparar los resultados con los métodos anteriores.

```
pred.ppr <- predict(ppreg, newdata = test)
obs <- test$Sqrt_Price

#Gráfico
pred.plot(pred.ppr, obs, xlab = "Predicción", ylab = "Observaciones")
```



Aquí se aprecia que las rectas de los modelos lineales y lm se acercan muy bien a la recta $y = x$ en valores de Predicción bajos

```
#Medidas de Error
(acc_3 <-accuracy(pred.ppr, obs))
```

```
##         me       rmse        mae        mpe       mape  r.squared
## -4.7931505 45.2391044 34.1780520 -2.3223007  8.5455634  0.7425014
```

Con este modelo explicaríamos un 74% de la variabilidad del Sqrt_Price en nuevas observaciones.

De los errores podemos interpretar:

- RMSE: Es una medida de la magnitud promedio del error (es decir, las diferencias entre los valores observados y los predichos). En este caso, un RMSE de 45.24 sugiere que las predicciones se desvían de los valores observados en promedio por alrededor de 45 unidades.
- MAE: Es el promedio de los valores absolutos de los errores. Aquí, el modelo tiene un error promedio absoluto de 34.18 unidades, lo que indica una desviación media moderada en las predicciones.
- MPE: Calcula el error promedio en porcentaje, indicando si las predicciones están por encima o por debajo de los valores observados en términos relativos. Un valor negativo indica subestimación, por lo que el modelo tiende a predecir un 2.32% menos que los valores reales, en promedio.
- MAPE: Es el error porcentual absoluto promedio. En este caso, el error absoluto promedio es del 8.55%, lo cual puede ser considerado como una precisión aceptable dependiendo del contexto y del rango de los valores observados.
- R-SQUARED: Mide la proporción de la varianza en los valores observados que es explicada por el modelo. Un R-SQUARED de 0.74 significa que el modelo explica aproximadamente el 74% de la

variación en los datos observados, lo que sugiere que el modelo captura una gran parte de la relación entre las variables predictoras y el resultado, pero no todas.

Con un rsquared de 0.76, el segundo modelo es el que mejor explica la variabilidad del Sqrt_Price en nuevas observaciones, seguido del tercer modelo con un rsquared de 0.74 y finalmente el primer modelo con un rsquared de 0.72.