Almacenes y Minería de Datos

Evaluación y selección de modelos

Evaluación y selección de modelos

- **Evaluación de modelos**: ¿es un modelo bueno? ¿el modelo lineal es razonable? ¿una regresión logística clasifica bien?
- **Selección de modelos**: ¿Existe un subconjunto de predictores que explica bien el modelo? ¿Puedo elegir un modelo mejor?
- Explicaremos técnicas para responder a estas preguntas.
- En general, estás técnicas se podrán usar tanto para evaluar como para seleccionar modelos.

Evaluación de modelos

- Ajustamos un modelo con nuestros datos de entrenamiento: minimizamos el error de entrenamiento.
- ¿El modelo se comportará bien para datos no vistos?
- Nos interesa que el error de test sea pequeño.
- ¿Cómo estimar el error de test?

Evaluación de modelos: conjunto de validación

- **Conjunto de validación**: dividimos el conjunto de datos disponible en dos partes.
- Una parte para ajustar el modelo: conjunto de entrenamiento.
- Otra parte para validar el modelo: conjunto de validación.
- Normalmente: 80 % entrenamiento, 20 % validación.
- Visto en el tema anterior.

R: conjunto de validación

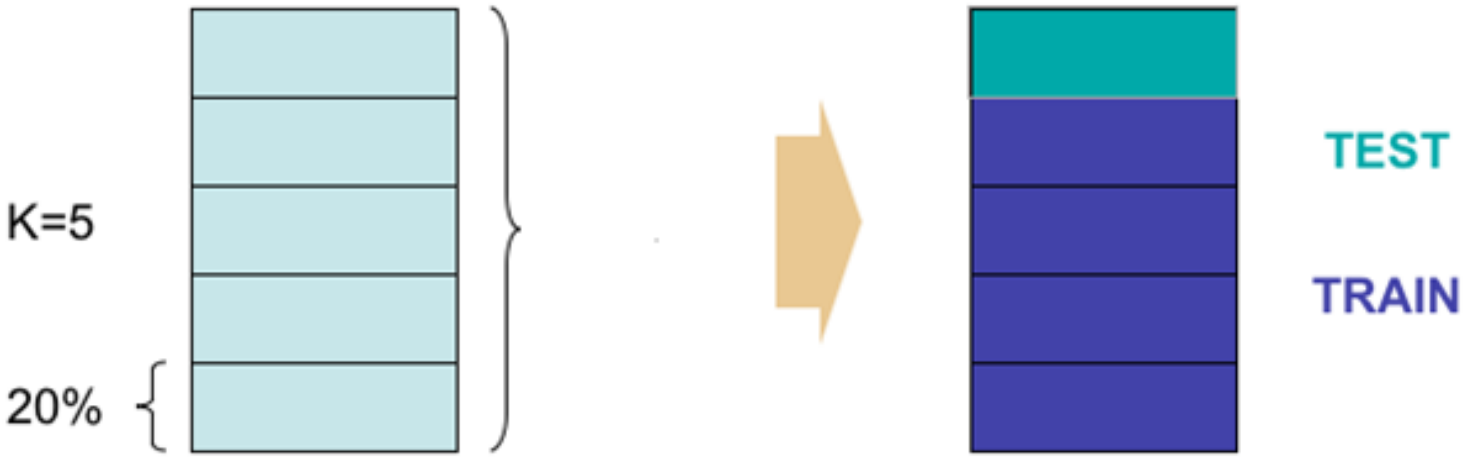
```
smk.data = read.table("birthsmokers.txt",header=T)
#elegir al azar 0.8*nrow(smk.data) números entre 1:nrow(smk.data)
 ind.train =sample(1:nrow(smk.data), 0.8*nrow(smk.data))
 smk.data.train =smk.data[ind.train ,]
 smk.data.test =smk.data[-ind.train ,]
smk.model.train=glm(Smoke~Gest+Wgt,data=smk.data.train,
                                          family="binomial")
 test.pred =predict(smk.model.train, smk.data.test ,  type="response")
  test.clasf =ifelse( test.pred >=0.5,"yes","no")
table( test.clasf , smk.data.test $Smoke)
 test.clasf   no yes
         no    4    1
         yes   0    2
```

Error en el conjunto de test: $1/7$

Conjunto de validación: problemas

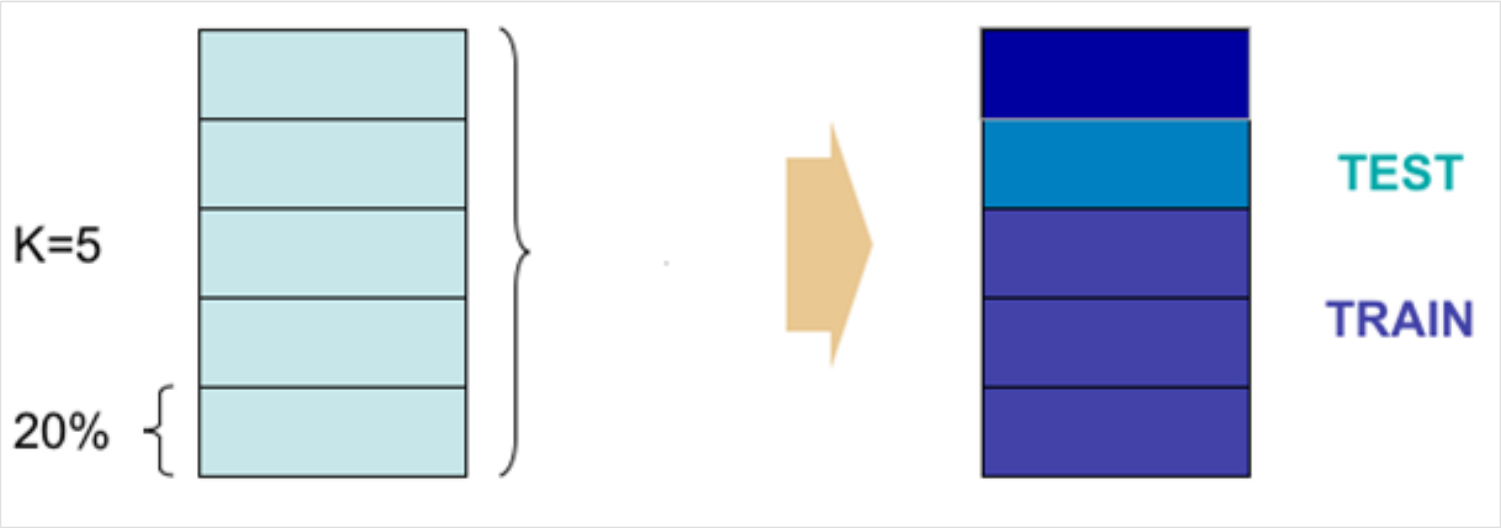
- El error de test calculado mediante el conjunto de validación puede ser muy variable. Depende del subconjunto elegido para validar.
- Solo se utiliza un subconjunto para ajustar el modelo.
- Solución: utilizar todos los datos como entrenamiento y test, **validación cruzada**

Validación cruzada



- Dividimos el conjunto de datos en k grupos del mismo tamaño.
- Conjunto de test: primer grupo. Conjunto de entrenamiento: resto de grupos
- Ajustamos el modelo con los datos de entrenamiento. Hallamos error con el conjunto de test
  - Problema regresión: con mínimos cuadrados obtenemos MSE_1, media de todos errores cometidos al cuadrado.
  - Problema clasificación: total de individuos mal clasificados Err_1

Validación cruzada



- Conjunto de test: segundo grupo. Conjunto de entrenamiento: resto de grupos
- Ajustamos el modelo con los datos de entrenamiento. Hallamos error con el conjunto de test
  - Problema regresión: MSE_2
  - Problema clasificación: Err_2

Validación cruzada

- Calculamos para cada grupo el correspondiente error:
  - Problema regresión: $MSE_1, MSE_2, \cdots, MSE_k$
  - Problema clasificación: $Err_1, Err_2, \cdots, Err_k$
- Finalmente, error estimado de test sería la media de los k errores calculados anteriormente:

$$\text{Problema regresión:} \qquad CV_{(k)} = \frac{\sum_1^k MSE_i}{k}$$

$$\text{Problema clasificación:} \qquad CV_{(k)} = \frac{\sum_1^k Err_i}{k}$$

Validación cruzada dejando uno fuera (LOOCV)

- Caso particular de validación cruzada: $k = n$, n igual al total de datos.
- Cada grupo está formado por una única observación.
- Se ajusta el modelo con todos los datos dejando uno fuera; test con un único dato.

$$CV_{(n)} = \frac{\sum_{i=1}^{n} MSE_i}{n}$$

$$CV_{(n)} = \frac{\sum_{i=1}^{n} Err_i}{n}$$

- Ventajas: hallar CV_k tiene fórmula cerrada para algunos casos (por ejemplo, regresión lineal); siempre los mismos resultados.
- Desventaja: si no hay fórmula cerrada, se tiene que ajustar el modelo n veces. Si n grande, muy caro.

R: validación cruzada regresión lineal

```
#Cargar librería boot para utilizar cv.glm
library(boot)
adv.data = read.csv(" Advertising.csv ")
#ajustar modelo lineal glm( ... family="gaussian"):equivalente a lm( .. )
adv.model = glm(Sales~TV+Radio, data=adv.data, family = "gaussian")
#validación cruzada con k=10
adv.cv = cv.glm(adv.data, adv.model, K=10)
print(adv.cv$delta [1])

[1] 2.911489
```

$$CV_{(10)} = 2.911489$$

Error estimado (MSE): 2.911489

R: validación cruzada regresión logística

```
#Cargar librería boot para utilizar cv.glm
library(boot)
smk.data = read.table("birthsmokers.txt", header = T)
#ajustar modelo
adv.model = glm(Smoke~., data=smk.data, family = "binomial")
#validación cruzada con k=10
smk.cv = cv.glm(smk.data, smk.model, K=10)
print(smk.cv$delta[1])

[1] 0.1409565
```

$$CV_{(10)} = 0.1409565$$

Error estimado: 14 % mal clasificado

R: LOOCV

- Para LOOCV: usar valor por defecto de K

```
#LOOCV: K igual a total de datos
smk.cv = cv.glm(smk.data, smk.model)
print(smk.cv$delta[1])

[1]  0.1429154
```

$$CV_{(32)} = 0.1429154$$

Selección de modelos: selección predictores

- Tenemos muchos predictores: ¿todos influyen en la respuesta? ¿Podemos simplificar el modelo?
- Idea simple:
  - Ajustar el modelo con distintos subconjuntos de predictores.
  - Quedarnos con el "*mejor*" modelo.
- Dos problemas:
  - ¿Cuándo un modelo es "*mejor*" que otro?
  - Si tenemos p predictores: inviable probar 2^p modelos con p grande.

¿Qué significa "*mejor*" modelo?

- ¿Cuándo podemos decir que un modelo es mejor que otro?
- Hemos visto algunos criterios para decidir qué modelo es mejor:
  - Regresión lineal: mayor R^2 (o equivalentemente menor error cuadrático medio).
  - Regresión logística: menor deviance.
- Pero con estos criterios un modelo con mas predictores tiene mejor resultado.
- Aunque los predictores sean espúreos.

Otros criterios

- Validación cruzada para estimar el error: puede suponer ajustar muchos modelos.
- Otros criterios que penalizan modelos con más predictores (pero suponen condiciones al modelo):
  - AIC: Aikake information criterion, mejor cuanto más bajo.
  - BIC: Bayesian information criterion, mejor cuanto más bajo.
  - C_p: Mallow's criterion, mejor cuanto más bajo (solo para mínimos cuadrados).
  - R^2 ajustado: mejor cuanto más grande.

Método: Mejor subconjunto (best subset)

Supone ajustar todos los modelos posibles con p predictores.

1. Empezamos con M_0 el modelo sin predictores.
2. Para $i = 1$ hasta p
   ajustar todos los modelos con i predictores y quedarnos con el mejor modelo, M_i:
   - Regresión lineal: mejor R^2.
   - Regresión logística: mejor deviance.
3. Entre los modelos $\{M_0, M_1, \cdots, M_p\}$ quedarnos con el mejor según algún criterio: validación cruzada, AIC, BIC, C_p, R^2-ajustado.

Problema

Información sobre la producción agrícola en 22 países[1]:

- País
- Producción agrícola (Millones dólares)
- Población activa en Agricultura (miles)
- Tierras cultivables (miles de acres)
- Ratio de conversión de pasto de tierra cultivable
- Ganadería Productiva (miles de animales)
- Stock de trabajo (miles de unidades)
- Consumo de fertilizantes(miles de tonelada métricas)
- Número de tractores

¿Qué predictores son los mejores para explicar la producción agrícola?

[1] Disponible en `http://www.stat.ufl.edu/~winner/data/worldagprod.dat`

R: best subset regresión lineal, R^2-ajustado, BIC, C_p

```
library(leaps)
agr.data = read.csv("worldagrprod.dat")
#best subset para regresión lineal con regsubsets
 agr.fit.best = regsubsets(agr.output~.-country, agr.data)
agr.summ = summary(agr.fit.best)
print(agr.summ)
##Ver salida en R
> agr.summ$bic
[1] -51.83 -69.95 -72.14 -76.14 -74.42 -72.54 -69.44
> agr.summ$adjr2
[1] 0.92 0.96 0.97 0.9807 0.9808 0.9806 0.979
> agr.summ$cp
[1] 54.4 11.61 7.72 3.73 4.78 6.05 8.0
> coef( agr.fit.best , 4) #coefs del mejor modelo con 4 predictores
(Intercept) pop.act.agr arables.land work.stock fert.consump
 62.9538117 0.1050454 0.0238893 -0.1108077 1.3090270
```

R: best subset regresión lineal, CV, AIC

```
library (bestglm)
#preparar datos: quedarnos en X con predictores
X=agr.data[,3:ncol(agr.data )]
#preparar datos: quedarnos en y con respuesta
y=agr.data [,2]
Xy=cbind(X,y)
 best.fits =bestglm(Xy,family=gaussian,IC="CV", #IC="AIC"
                                CVArgs=list(Method="HTF",K=5,REP=1))
print( best.fits )
  Best Model:
                    Estimate    Std. Error    t value      Pr(>|t|)
  (Intercept)  253.91192207 1.653607e+02 1.535503 1.411455e-01
   arables.land   0.02550582 1.795012e-03 14.209276 1.421307e-11
  fert.consump   1.34328471 2.419465e-01 5.551991 2.352727e-05

print( best.fits $Subsets); print( best.fits $BestModel)#ver salida en R
```

R: best subset regresión logística

```
library (bestglm)
smk.data = read.table(" birthsmokers.txt ")
X = smk.data[,1:2]
y = smk.data[,3]
Xy=cbind(X,y)
#añadir parámetro: IC="AIC"; IC="CV", CVArgs=igual transp. anterior
print( bestglm(Xy, family = binomial))
BIC
Best Model:
                    Estimate Std. Error    z value     Pr(>|z|)
(Intercept) -52.63571385 19.7695783 -2.662460 0.007757176
Wgt          -0.01961458 0.0074627 -2.628349 0.008580040
Gest          2.89682530  1.0670198  2.714875 0.006630080
```

Selección: muchos predictores

- Si p grande, imposible ajustar 2^p modelos.
- Se utiliza algún tipo de algoritmo voraz para conseguir el "*mejor*" modelo.
- Regresión paso a paso: incorporando predictores (forward), eliminando predictores (backward)

Regresión paso a paso: forward

1. Empezamos con M_0 el modelo sin predictores.
2. Para $i = 0$ hasta $p - 1$

   Generar todos los modelos añadiendo solo un predictor de los no usados en M_k. Entre estos modelos quedarse con aquel M_{k+1}:
   - Regresión lineal: mejor R^2.
   - Regresión logística: mejor deviance.
3. Entre los modelos $\{M_0, M_1, \cdots, M_p\}$ quedarnos con el mejor según algún criterio: validación cruzada, AIC, BIC, C_p, R^2-ajustado.

Regresión paso a paso: backward

1. Empezamos con M_p el modelo con todos los predictores.
2. Para $i = p$ hasta 1

   Generar todos los modelos eliminando solo un predictor de los usados en M_k. Entre estos modelos quedarse con aquel M_{k-1}:
   - Regresión lineal: mejor R^2.
   - Regresión logística: mejor deviance.
3. Entre los modelos $\{M_0, M_1, \cdots, M_p\}$ quedarnos con el mejor según algún criterio: validación cruzada, AIC, BIC, C_p, R^2-ajustado.

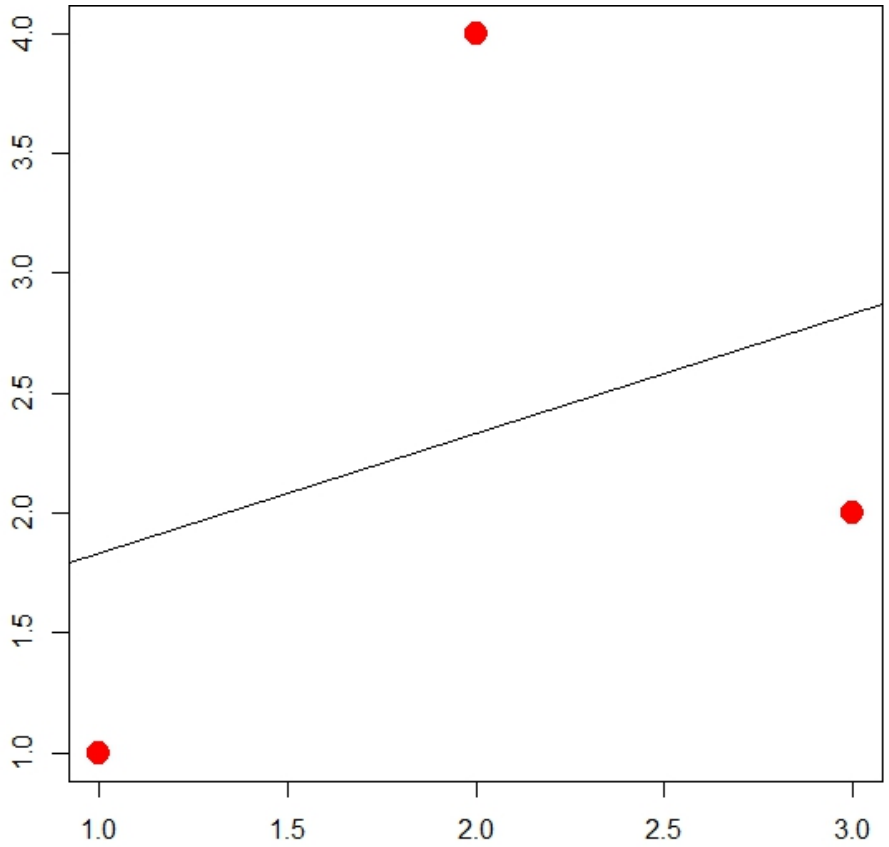
R: regresión paso a paso

```
library (bestglm)
 ....
best.fw=bestglm(Xy,family=gaussian,IC="AIC",method="forward")


 ....
best.bk=bestglm(Xy, family = binomial, method = "backward")
```

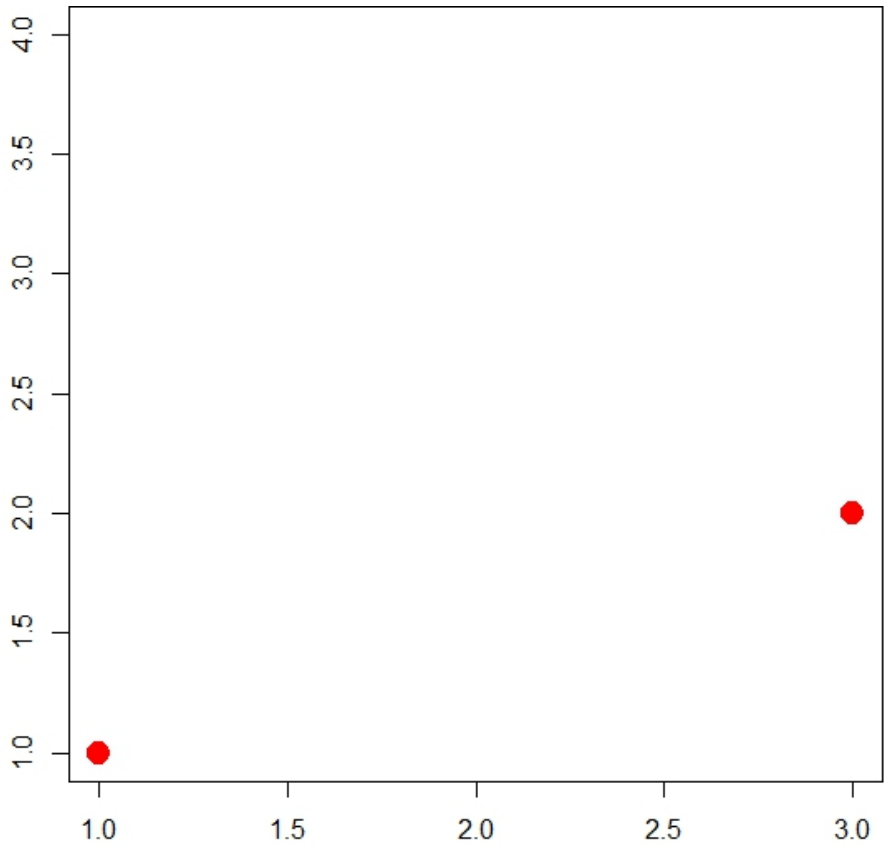
Regresión Ridge y Lasso

- Problema con un predictor X_1 y tres observaciones, $n = 3$. Regresión lineal:



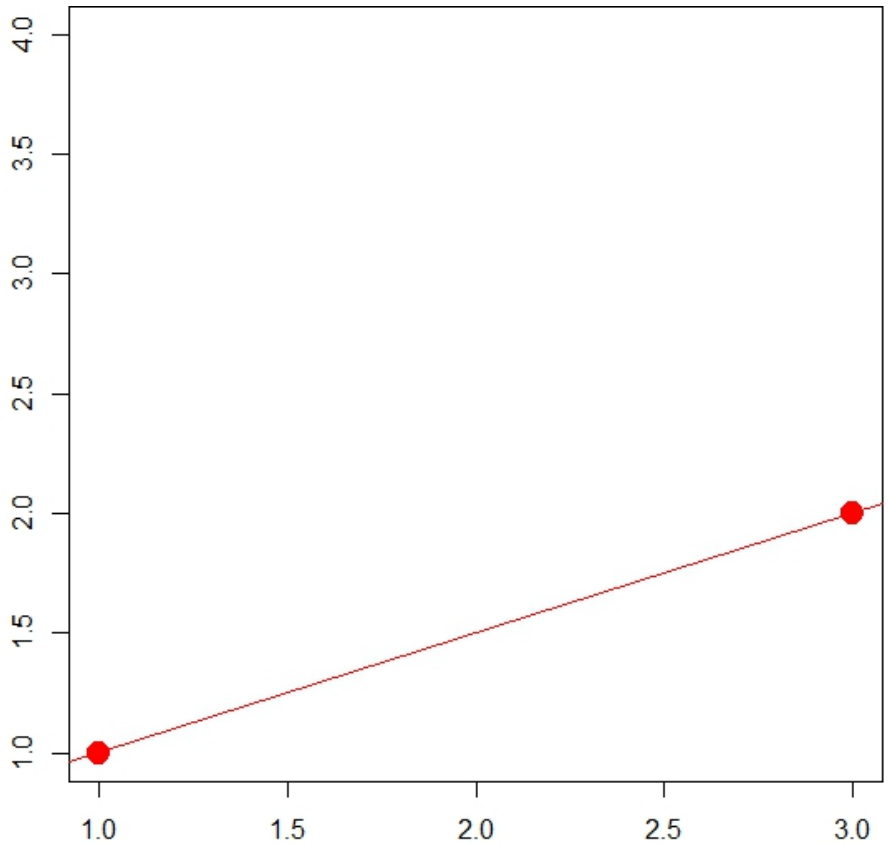
Regresión Ridge y Lasso

- Eliminamos una observación: $n = 2$. ¿Cuál es la recta de regresión?



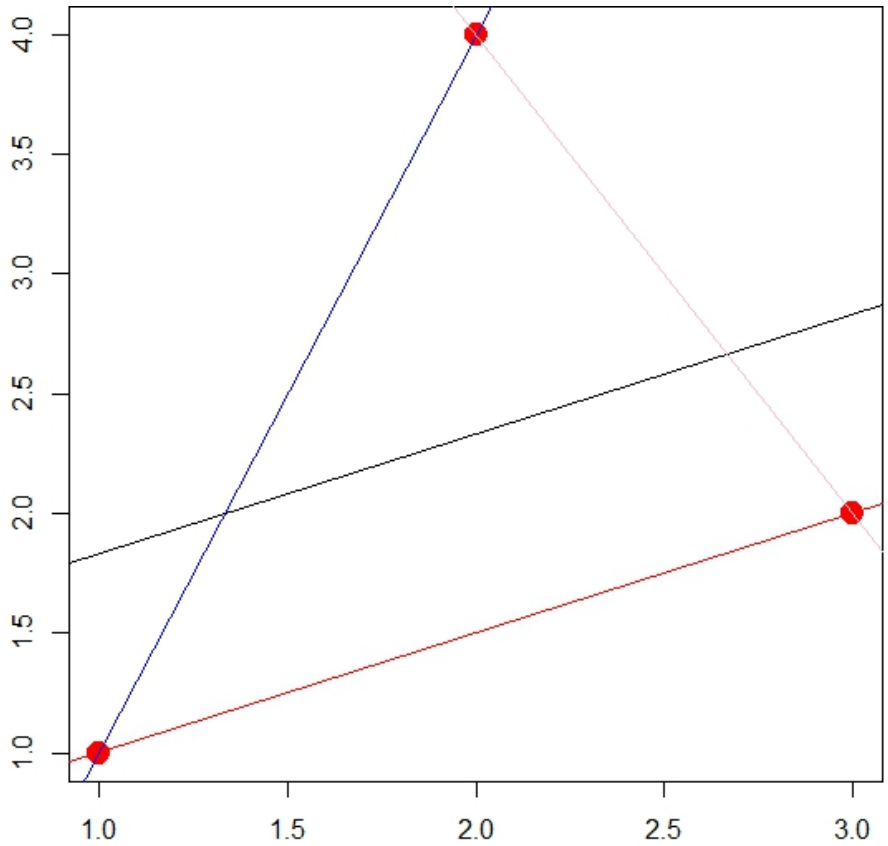
Regresión Ridge y Lasso

- Eliminamos una observación: $n = 2$. ¿Cuál es la recta de regresión?



Regresión Ridge y Lasso

- Si $p \gg n$: mucha varianza, poco bias.



Regresión Ridge y Lasso

- Recordad, para la regresión lineal minimizamos (mínimos cuadrados):

$$RSS = \sum_{i=1}^{n} \left(y_i - (\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij})\right)^2 = L$$

- En la regresión logística maximizamos:

$$\prod_{y_i=1} \frac{1}{1+e^{-(\beta_0+\sum_{j=1}^{p}\beta_j x_{ij})}} \prod_{y_i=0} \left(1 - \frac{1}{1+e^{-(\beta_0+\sum_{j=1}^{p}\beta_j x_{ij})}}\right)$$

o equivalentemente, minimizar

$$L = -log(\prod_{y_i=1} \frac{1}{1+e^{-(\beta_0+\sum_{j=1}^{p}\beta_j x_{ij})}} \prod_{y_i=0} \left(1 - \frac{1}{1+e^{-(\beta_0+\sum_{j=1}^{p}\beta_j x_{ij})}}\right))$$

Regresión Ridge y Lasso

Se introduce un **término de regularización** para penalizar coeficientes β_i altos.

- Regresión Ridge, se minimiza

$$L + \lambda \sum_{k=1}^{p} \beta_k^2$$

- Regresión Lasso

$$L + \lambda \sum_{k=1}^{p} |\beta_k|$$

- Los coeficientes β_i tenderán a ser bajos.
- Objetivos:
  - **Mejorar el modelo**: cuando tenemos $p \gg n$.
  - **Seleccionar predictores**: predictores con $\beta_i \neq 0$.
- **Problema**: se debe elegir correctamente el parámetro λ.

Regresión Ridge y Lasso

- Regresión Ridge

$$L + \lambda \sum_{k=1}^{p} \beta_k^2$$

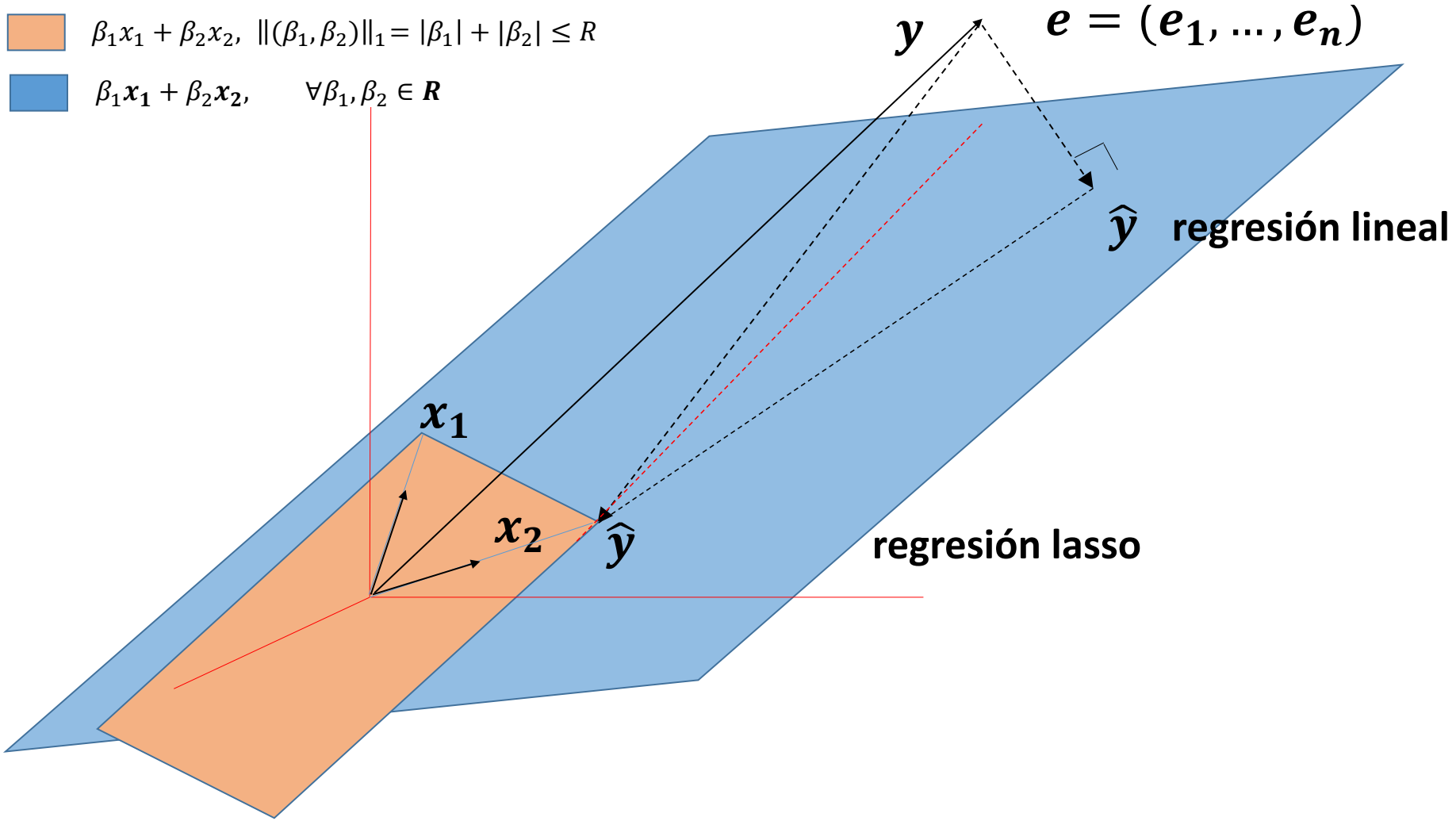
$$minimizar(L)\ sujeto\ a\ \sum_{i=1}^{p} \beta_i^2 < R$$

- Regresión Lasso

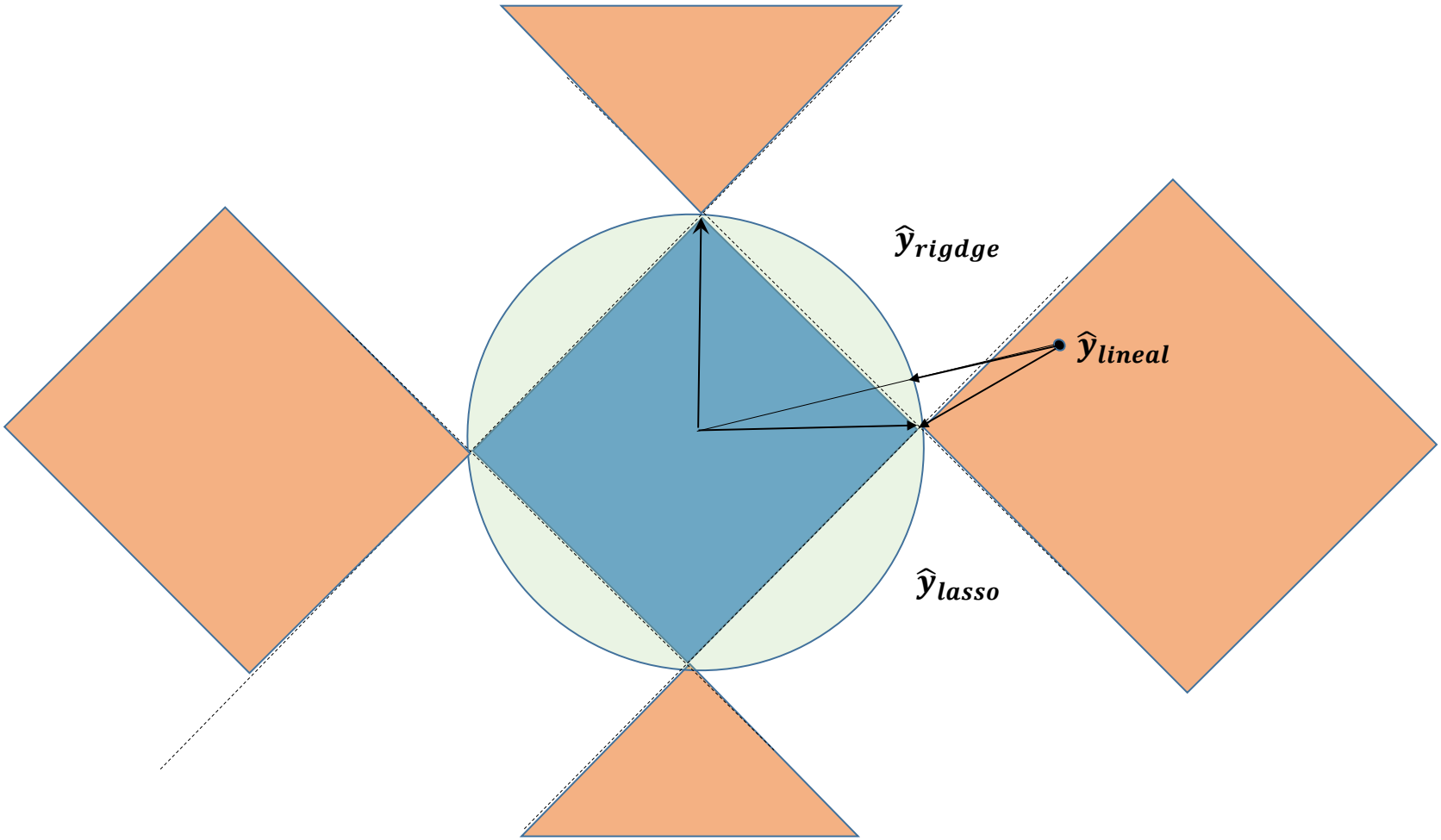
$$L + \lambda \sum_{k=1}^{p} |\beta_k|$$

$$minimizar(L)\ sujeto\ a\ \sum_{i=1}^{p} |\beta_i| < R$$

Efecto de usar $||\boldsymbol{\beta}||_2 = \sum_{i=1}^{p} \beta_i^2$ o $||\boldsymbol{\beta}||_1 = \sum_{i=1}^{p} |\beta_i|$



Efecto de usar $||\boldsymbol{\beta}||_2 = \sum_{i=1}^{p} \beta_i^2$ o $||\boldsymbol{\beta}||_1 = \sum_{i=1}^{p} |\beta_i|$



Regresión Ridge y Lasso

Observad que en el modelo lineal ajustado con mínimos cuadrados

- si el coeficiente estimado para el predictor X_i es $\hat{\beta}_i$;
- si cambio X_i por $Y_i = X_i/b$, el coeficiente estimado para Y_i es $b\hat{\beta}_i$.
- Esto es, es indiferente las unidades en la que está medido el predictor
- En las regresiones Ridge y Lasso no ocurre lo mismo al introducir el término de regularización
- Aconsejable tipificar los predictores: usar $yi = (x_i - \bar{x})/sd(x)$

Regresión Ridge y Lasso

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p + \epsilon \qquad \text{Modelo regresión lineal}$$

$$P(Y = 1|X = x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p)}} \qquad \text{Modelo regresión logística}$$

- Probamos para varios valores de λ: por ejemplo, $\lambda = 0.01..10^{10}$
- Para cada valor de λ, ajustamos el modelo M_λ mediante:

$$L + \lambda \sum_{k=1}^{p} \beta_k^2 \qquad \text{regresión Ridge}$$

$$L + \lambda \sum_{k=1}^{p} |\beta_k| \qquad \text{regresión Lasso}$$

Regresión Ridge y Lasso

- Estimamos el error de test para cada modelo M_λ: validación cruzada.
- Nos quedamos con el "*mejor*" modelo M_{λ_0}
- Estimamos error de test del mejor: M_{λ_0}

R: regresión Ridge y Lasso

Ver fichero lassoBinomial.R