Análisis Avanzado de Datos.

Nicolás López

Primer semestre de 2023

Recordando definiciones iniciales

Introducción

Aprender el proceso subyacente generador de datos. El proceso es formalizado matemáticamente en el aprendizaje estadístico y se clasifica en dos grupos:

- Aprendizaje supervisado: Se tiene un resultado (*outcome*) que guía el proceso de aprendizaje (ej. identificación de dígitos).
- Aprendizaje no supervisado. No se tiene una medición de un resultado para guiar el aprendizaje (ej. clasificación de carros basado en características).

En ambos escenarios se cuenta con un conjunto de covariables (*features*) que permiten el aprendizaje.

Definiciones importantes

Se destacan 4 elementos fundamentales en el aprendizaje estadístico dada la revisión anterior:

- Proceso generador P.
- Variable de entrada/covariable/input: X (uni/multivariada).
- Variable de salida/variable respuesta/output: Y (univariada usualmente).
- Observaciones/realizaciones/mediciones: $(x_1, y_1), ..., (x_n, y_n)$.

Y se recuerdan 3 elementos adicionales de la sesión anterior:

- Para RLS/RLM β_{MCO} es el BLUE, lo cual no es necesariamente bueno.
- $$ECM(\hat{\theta}) = V(\hat{\theta}) + (E(\hat{\theta}) - \theta)^2 = V(\hat{\theta}) + B(\hat{\theta})^2$$
- El sesgo λ de la estimación ridge/lasso se selecciona de forma que minimice el error en un **conjunto de datos de prueba**.

Muestreo. ¿por qué muestrear de una población?

En estadística supervisada clásica se obtienen $(x_1, y_1), ..., (x_n, y_n)$ generadas por un proceso P, generalmente indexado por $\theta \in \Theta$. Estas n observaciones son utilizadas para estimar θ:

1. Si el interés es interpretabilidad del problema, este acercamiento permite caracterizar el fenómeno mediante $\hat{\theta}$: se validan supuestos / prueban hipótesis / ... : **validación interna** (no requiere datos externos).
2. **Evaluación del modelo**: Si el interés es predicción de nuevas observaciones, requerimos un conjunto de prueba para medir el rendimiento de $P_{\hat{\theta}}$: **validación externa**.

Sin embargo, en regresión penalizada notamos que el sesgo λ es un **hiperparámetro** a ser seleccionado. Por lo cual se abre una tercera alternativa:

3. **Selección del modelo** Si el interés es seleccionar un modelo entre varias alternativas: (por ejemplo, $\lambda > 0$ en el modelo regularizado), requerimos un conjunto de prueba para medir el rendimiento de $P_{\hat{\theta}_1,\lambda_1},\ldots,P_{\hat{\theta}_l,\lambda_l}$: **validación externa**.

En (2) y (3) es difícil contar con un conjunto externo de datos de prueba, por lo cual hay dos alternativas:

- Confiar en el conjunto de **datos de entrenamiento**: usar el error de entrenamiento.
- Dejar afuera parte de nuestros datos (particionarlos): extraer una muestra de entrenamiento y una muestra de prueba.

Usualmente, para evaluar el rendimiento de un modelo estadístico (2) o seleccionar el mejor modelo entre varias alternativas (3) se utiliza el error cuadrático medio (**de predicción**, note la diferencia con $ECM(\hat{\beta})$). En el caso de un análisis de regresión este está dado por:

$$ECM(\hat{f}) = \frac{\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2}{n}$$

Dónde $\hat{f}$ es la función estimada dados los datos. La validación interna da lugar a $ECM(\hat{f})_{Train} = ECM_{Train}$ y al externa a $ECM(\hat{f})_{Test} = ECM_{Test}$.

Note que la validación interna puede llevarse a cabo mediante validación externa: si tengo dos modelos $\hat{f}_1$ (completo) y $\hat{f}_0$ (reducido), puedo medir su significancia mediante una prueba estadística (1) o evaluar la diferencia en $ECM(\hat{f}_1)$ y $ECM(\hat{f}_0)$ (2).

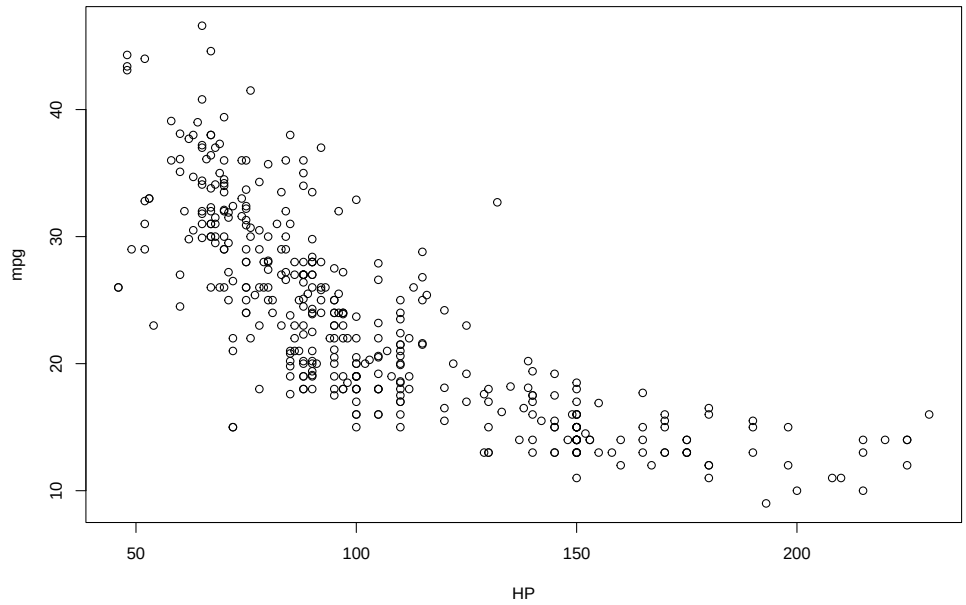
Hay múltiples formas de validar externamente un modelo

- Validación regular.
- Validación cruzada dejando un individuo afuera.
- Validación cruzada en k **folds**.

Validación regular

Evidentemente la relación entre las variables es no lineal en la variable *hp*. Pero, ¿qué potencia debería usar?

```
library('ISLR2') ; library('glmnet')
plot(Auto$horsepower,Auto$mpg,xlab="HP",ylab="mpg")
```



Este es un problema de **selección del modelo**.

La validación regular toma una división generalmente homogénea de *train* y *test* para un mismo conjunto de datos, y con esto, compara el *ECM* de los modelos resultantes:

```
sample_size <- nrow(Auto)
set.seed(456)
train       <- sample(sample_size, 0.5*sample_size)
test        <- seq(sample_size)[!seq(sample_size) %in% train]

mod1 = lm(mpg ~ horsepower,data=Auto[train,])
mse1 = mean((Auto$mpg[test] - predict(mod1,Auto)[test])**2)
mse1
```

```
## [1] 22.27728
```

```
mod2 = lm(mpg ~  poly(horsepower,2),data=Auto[train,])
mse2 = mean((Auto$mpg[test] - predict(mod2, Auto)[test])**2)
mse2
```

```
## [1] 16.38132
```

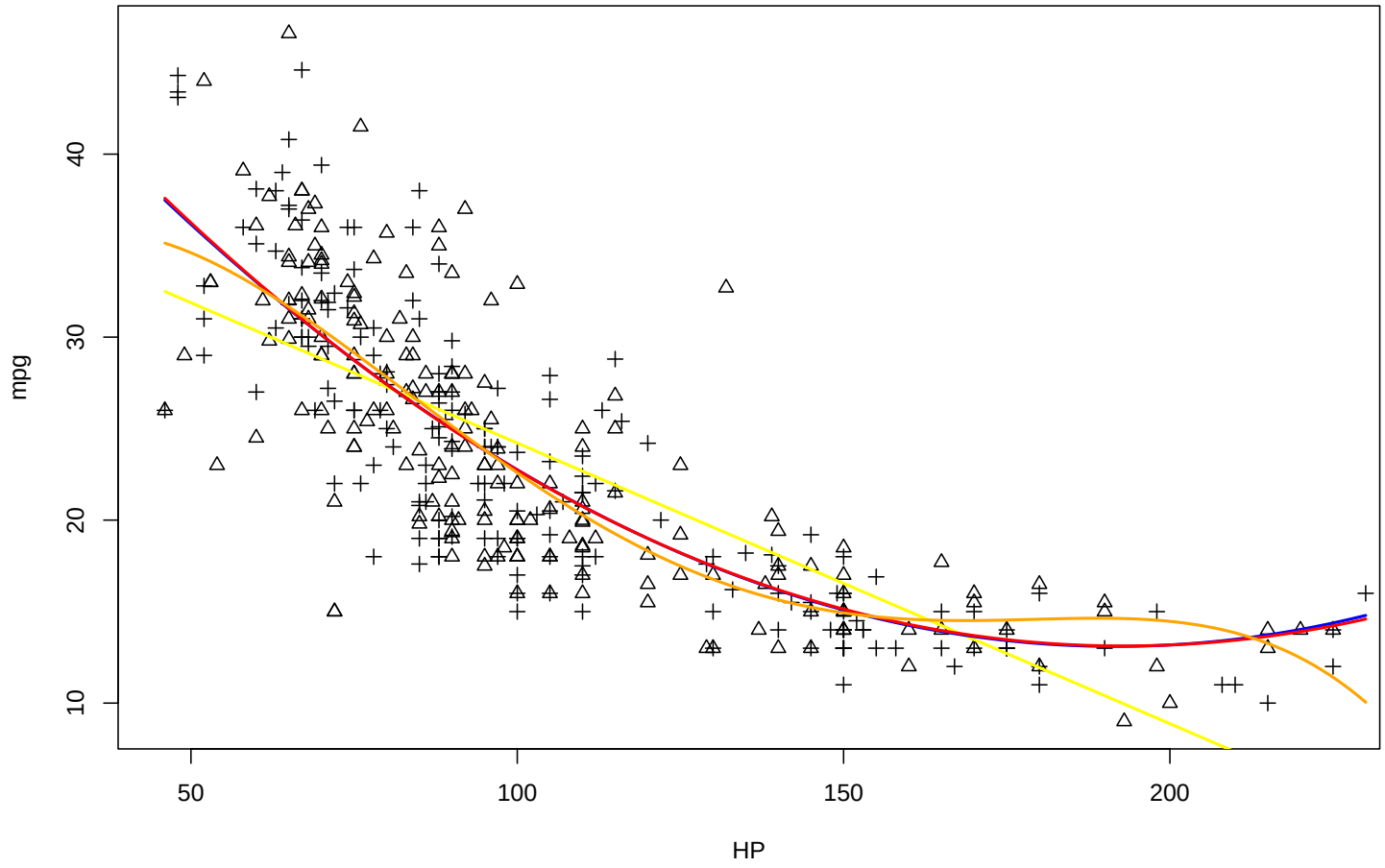
```
mod3 = lm(mpg ~  poly(horsepower,3),data=Auto[train,])
mse3 = mean((Auto$mpg[test] - predict(mod3, Auto)[test])**2)
mse3
```

```
## [1] 16.3591
```

```
mod4 = lm(mpg ~  poly(horsepower,4),data=Auto[train,])
mse4 = mean((Auto$mpg[test] - predict(mod4, Auto)[test])**2)
mse4
```

```
## [1] 16.8401
```

En efecto, el ajuste en *train* (triángulos) reflejado en *test* (positivos) no muestra que polinomos de grado mayor a dos sean significativos:



¿Cómo se realiza esta prueba estadística? (validación interna del modelo)

El modelo es ajustado y se observan los valores p correspondientes

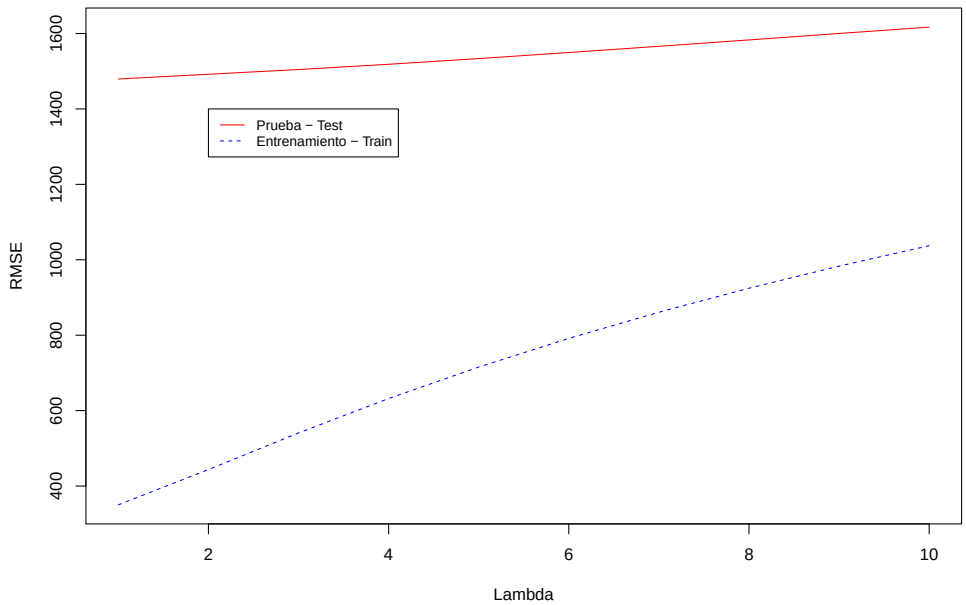
```
modt = lm(mpg ~  poly(horsepower,4),data=Auto)
summary(modt)
```

```
##
## Call:
## lm(formula = mpg ~ poly(horsepower, 4), data = Auto)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.8820  -2.5802  -0.1682   2.2100  16.1434
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              23.4459     0.2209 106.161   <2e-16 ***
## poly(horsepower, 4)1 -120.1377     4.3727 -27.475   <2e-16 ***
## poly(horsepower, 4)2   44.0895     4.3727  10.083   <2e-16 ***
## poly(horsepower, 4)3   -3.9488     4.3727  -0.903    0.367
## poly(horsepower, 4)4   -5.1878     4.3727  -1.186    0.236
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.373 on 387 degrees of freedom
## Multiple R-squared:  0.6893, Adjusted R-squared:  0.6861
## F-statistic: 214.7 on 4 and 387 DF,  p-value: < 2.2e-16
```

Note que no se particionan los datos, el modelo se ajusta sobre la tabla completa.

Este acercamiento permite observar dos características importantes en la medición del error bajo los dos conjuntos de datos. Tomemos como ejemplo ahora la indexación de modelos ridge en función de λ:

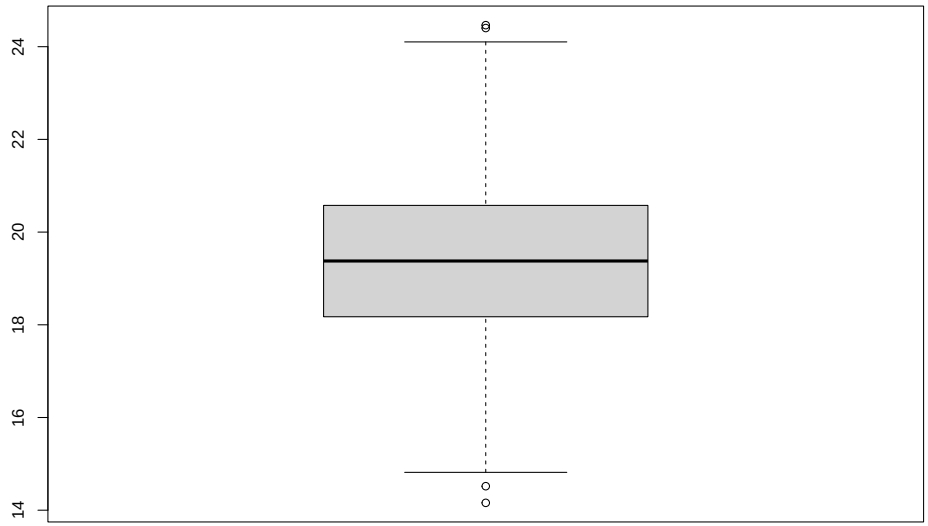
```
set.seed(123) ; grid        <- seq(10)
x           <- model.matrix(mpg ~ ., Auto)[, -1] ; y <- Auto$mpg
sample_size <- nrow(x)
train       <- sample(sample_size, 0.5*sample_size)
test        <- seq(sample_size)[!seq(sample_size) %in% train]
fit_ridge   <- glmnet(x[train,],y[train],alpha=0,lambda=grid)
rmse1      <- NULL ; rmse2      <- NULL

for(i in 1:length(grid)){
rmse1[i] <- sqrt(crossprod(predict(fit_ridge,s = grid[i],newx=x[train,]) - y[train]) / 0.5*sample_size)
rmse2[i] <- sqrt(crossprod(predict(fit_ridge,s = grid[i],newx=x[test,]) - y[test]) / 0.5*sample_size)
}

plot(grid,rmse2,col="red",type='l',ylim=c(min(rmse1),max(rmse2)),xlab ="Lambda",ylab="RMSE")
lines(grid,rmse1,type='l',lty=2,col="blue")
legend(2, 1400,legend=c("Prueba - Test", "Entrenamiento - Train"),col=c("red", "blue"),lty=1:2, cex=0.8)
```



- Usualmente $ECM_{Train} << ECM_{Test}$.
- Un bajo ECM_{Train} puede ser alcanzado a costo potencial de aumentar ECM_{Test}. Esto es llamado sobreajuste u *overfitting*.

Volviendo al caso de RLS. Como observamos, la validación regular permite medir el error en un conjunto de prueba, sin embargo:

```
sample_size = nrow(Auto)
mse_sample  = NULL
for(i in 1:1000){
  set.seed(i)
  train          = sample(sample_size, 0.5*sample_size)
  test           = seq(sample_size)[!seq(sample_size) %in% train]
  mod_lm         = lm(mpg ~  poly(horsepower,2),data=Auto[train,])
  mse_sample[i] = mean((Auto$mpg[test] - predict(mod_lm, Auto)[test])**2)
}
boxplot(mse_sample)
```



¿Nota algún problema?

Hay dos problemas principales con la validación regular:

- Alta variabilidad en la estimación de ECM.
- Alto sesgo: La mitad de los datos disponibles son usados para ajustar el modelo -> sobreestimación de ECM.

Validación cruzada dejando un individuo afuera (LOOCV)

Validación cruzada dejando un individuo afuera (LOOCV)

Este esquema de muestreo para la validación externa toma la i-ésima observación como conjunto de *test* y las restantes como conjunto de *train*::

- Se tienen n en lugar de un ECM_{Test}: $ECM_{Test,1} = (y_i - \hat{f}_{-1}(x_i))^2, \ldots, ECM_{Test,n} = (y_n - \hat{f}_{-n}(x_i))^2$. Una estimación única de ECM_{Test} está dada por:

$$ECM_{Test} = \frac{\sum_{i=1}^{n} ECM_{Test,i}}{n}$$

Esto soluciona los dos problemas anteriormente mencionados a costo de un mayor consumo computacional, particularmente alto para modelos complejos.

- Tiene varianza 0: El proceso de muestreo no es aleatorio.
- Bajo sesgo: todos, menos un datos, son considerados en el entrenamiento: a través de las muestras hay alta concordancia en los elementos muestreados.

Para RLS/RLM, aún con componentes cuadráticos, el costo computacional es irrelevante, ya que basta ajustar un modelo para encontrar *ECM* de LOOCV. Sin embargo, esta es la excepción, y no la regla.

Puede calcularse manualmente

```
mse_i = NULL
for(i in 1:nrow(Auto)){
model_i  = lm(mpg ~ poly (horsepower,2), data = Auto[-i,])
mse_i[i] = (Auto$mpg[i] - predict(model_i, Auto[i,]))**2
}
mean(mse_i)
```

```
## [1] 19.24821
```

O mediante una función del programa

```
library(boot)
cv.error <- NULL
for (i in 1:4) {
glm.fit <- glm(mpg ~ poly (horsepower,i), data = Auto)
cv.error[i] <- cv.glm(Auto,glm.fit)$delta[1]
}
print(cv.error)
```

```
## [1] 24.23151 19.24821 19.33498 19.42443
```

Validación cruzada en k folds

Validación cruzada en k folds

Este esquema de muestreo para la validación externa toma una partición de tamaño k con componentes de igual dimensión. La i-ésima colección se toma como conjunto de *test* y las restantes como conjunto de *train*::

- Se tienen l en lugar de n o un ECM_{Test}: $ECM_{Test,1},\ldots,ECM_{Test,k}$. Una estimación única de ECM_{Test} está dada por:

$$ECM_{Test} = \frac{\sum_{i=1}^{k} ECM_{Test,i}}{k}$$

Es claro que si $k = n$ se tiene LOOCV.¿De qué depende la selección de k?

Suponga que tiene dos modelos candidatos, $M1$ y $M2$ y quere determinar mediante validación cruzada cuál de ellos seleccionar. Naturalmente seleccionará aquel con menor ECM_{Test}. Sin embargo, el valor de k diferencia el tipo de modelo seleccionado.

Si k tiende a n:

- El conjunto de entrenamiento es más grande - **menor sesgo** del modelo entrenado.
- El error de predicción es obtenido con n modelos entrenados casi con los mismos datos -> $ECM_{Test,i}$ con $i = 1, ...n$ correlacionados.
- ECM_{Test} es un promedio de cantidades muy correlacionadas -> **mayor varianza**,

Si k tiende a 1:

- El conjunto de entrenamiento es más pequeño - **mayor sesgo** del modelo entrenado.
- El error de predicción es obtenido con k modelos entrenados con pocos datos en común -> $ECM_{Test,i}$ con $i = 1, ...k$ poco correlacionados.
- ECM_{Test} es un promedio de cantidades poco correlacionadas -> **menor varianza**,

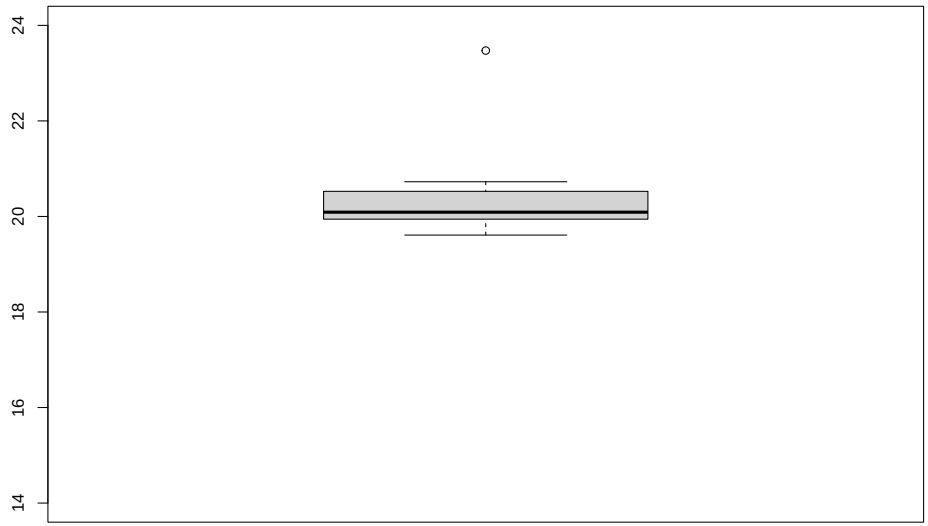
En general se toma $k = 5$ o $k = 10$.

Tenga en cuenta que esto sprocedimientos de validación externa fueron descritos para el escenario continuo. En el caso discreto (como la clasificación de una enfermedad), se adapta el *ECM* al caso discreto. Para LOOCV:

$$\frac{\sum_{i=1}^{n} I\{y_i \neq \hat{y}_i\}}{n}$$

Volviendo al caso de RLS. Como observamos, la validación cruzada permite medir el error en un conjunto de prueba:

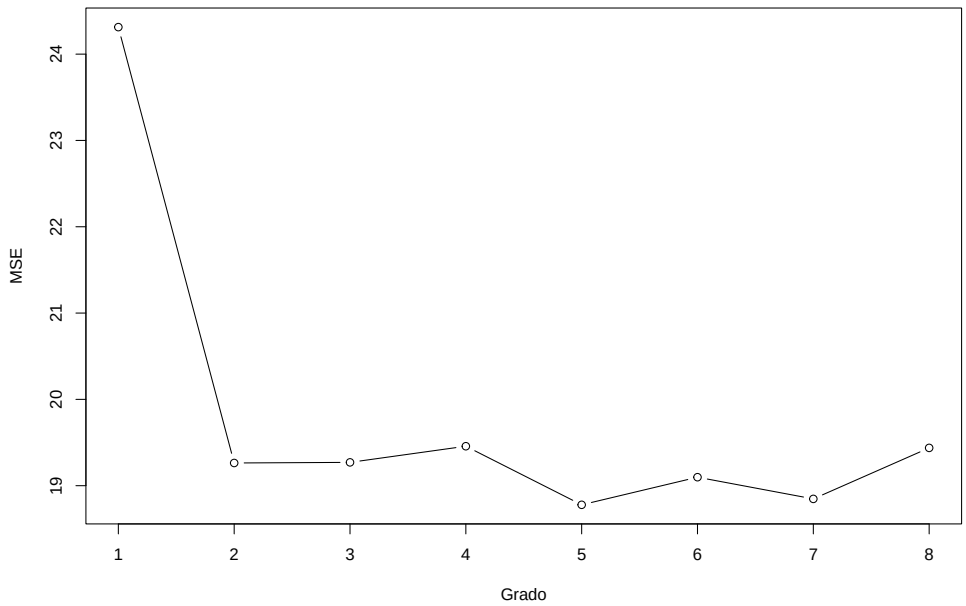
```
sample_size = nrow(Auto)
mse_sample  = NULL
set.seed(1)
rnd_sample  = sample(rep(1:10,length.out=sample_size))
for(i in 1:10){
  mod_lm        = lm(mpg ~  poly(horsepower,2),data=Auto[rnd_sample==i,])
  mse_sample[i] = mean((Auto$mpg[rnd_sample!=i] - predict(mod_lm, Auto)[rnd_sample!=i])**2)
}
boxplot(mse_sample,ylim=c(14,24))
```



La variabilidad es significativamente menor en el *ECM* respecto a LOOCV.

No es necesario realizar este cálculo manualmente, pues está implementado en R, sólo note la diferencia en el ajuste del modelo. Volviendo al ejemplo de regresión lineal politómica, podemos usar la validación externa para determinar la relevancia de grados superiores del polinomio:

```
set.seed(123)
cv.error.10 <- NULL
for (i in 1:8) {
glm.fit <- glm (mpg ~ poly (horsepower , i), data = Auto)
cv.error.10[i] <- cv.glm(Auto,glm.fit,K = 10)$delta[1]
}
plot(1:8,cv.error.10,xlab="Grado",ylab="MSE", type="b")
```



La regresión ridge y lasso usualmente hacen uso de validación externa con validación cruzada en 10 folds para determinar el valor de λ a seleccionar. No siempre es el λ_{min}, también se usa λ_{1se}.

Bootstrap

Bootstrap

Al tener una m.a $X_1,\ldots,X_n$ de observaciones normales su promedio es normal, con varianza σ^2/n. Sin embargo

1. Otros estadísticos no tienen una distribución cerrada, lo cual dificula la construcción de IC -> hipótesis.
2. Algunos estadísticos podrían ser aproximados mediante TLC (como las proporciones), pero no permite realmente una mayor flexibilidad para algunos casos de interés.

El proceso es simple. Supongamos que se desea la distribución de la **mediana** para una colección de n observaciones:

1. Se extrae una muestra de tamaño n *con remplazo* (con devolución).
2. Se calcula la **mediana** de la muestra.
3. Se almacena el valor y se vuelve al paso (1) hasta lograr un número determinado de repeticiones.

La distribución de los valores almacenados se toma como la distribución de la **mediana**, y con el eror estándar se cuantifica la incertidumbre asociada en la estimación.

En lugar de la mediana podríamos pensar en un método de aprendizaje estadístico (como la regresión ridge o lasso vistas en la clase pasada), para calcular la distribución de los parámetros estimados. Para el caso de RLS tenemos:

```
boot.fn <- function (data , index){
           coef(lm(mpg ~ horsepower , data = data , subset = index))}

set.seed(1)
boot.fn(Auto , sample (392, 392, replace = T))
```

```
## (Intercept)  horsepower
##  40.3404517  -0.1634868
```

```
boot.fn(Auto , sample (392, 392, replace = T))
```

```
## (Intercept)  horsepower
##  40.1186906  -0.1577063
```

```
boot.fn(Auto , sample (392, 392, replace = T))
```

```
## (Intercept)  horsepower
##  40.1834549  -0.1585993
```

Este proceso puede ser automatizado de la siguiente forma

```
boot.fn <- function (data , index){
           coef(lm(mpg ~ horsepower , data = data , subset = index))}

boot(Auto,boot.fn, 1000)
```

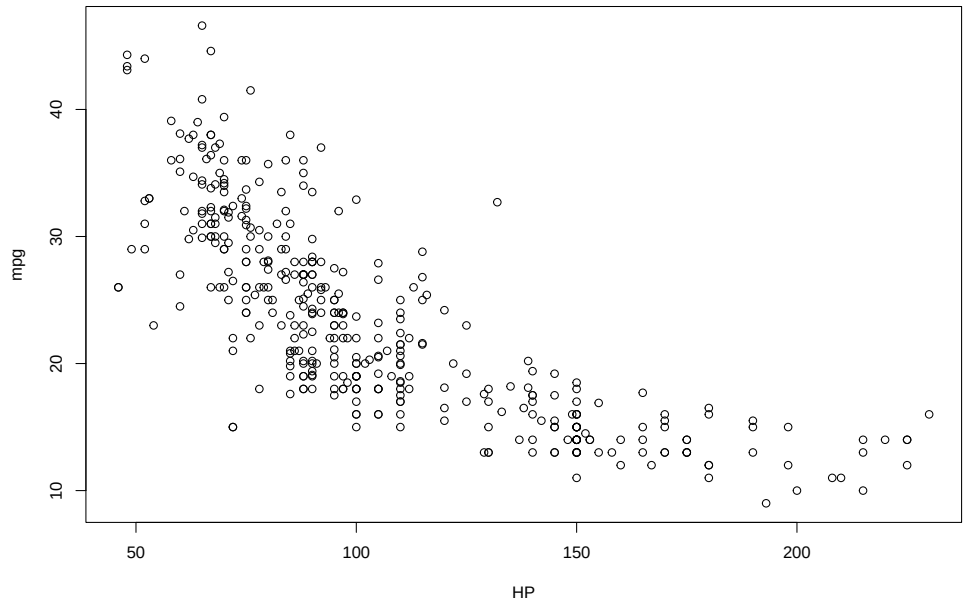
```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Auto, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##       original        bias    std. error
## t1* 39.9358610  0.0522971635  0.84001614
## t2* -0.1578447 -0.0006027039  0.00733243
```

Comparable con los resultados obtenidos de un modelo de regresión regular:

```
summary(lm(mpg ~ horsepower , data = Auto))$coef
```

```
##               Estimate  Std. Error   t value      Pr(>|t|)
## (Intercept) 39.9358610 0.717498656  55.65984 1.220362e-187
## horsepower  -0.1578447 0.006445501 -24.48914  7.031989e-81
```

Sin embargo existen ligeras diferencias entre los dos valores estimados. ¿Cuáles serán estimadores más precisos de los errores de los parámetros estimados? ¿Aquellos obtenidos mediante bootstrap o aquellos dados por el modelo ajustado?



Conclusiones importantes:

- Podemos artificialmente mejorar el ajuste en los datos de entrenamiento, pero no necesariamente esto garantiza un buen comportamiento en los datos de prueba. En dado caso, a esto se le llama **sobreajuste**.
- **Evaluación del modelo** - Los tres métodos vistos son de validación externa: buscan determinar qué tan bien se comportará el método en datos no vistos anteriormente.
- **Selección del modelo** - Podemos además observar que al seleccionar un modelo, es de interés el valor mínimo de la curva de error, mas no el valor del error en si mismo.

Conclusiones importantes:

- Si buscamos comprarar múltiples modelos entre si (3) que requieren encontrar uno o varios hiperparámetros (2). Por ejemplo, encontrar en un conjunto de datos determinado si ridge o lasso son mejores en la predicción. Usualmente es separado conjunto inicial de datos de **validación** para estimar el error de predicción en la selección del modelo (3).

Referencias

Referencias

1. Hastie, T., Tibshirani, R., Friedman, J. The Elements of Statistical Learning. Springer.
2. Garet, Witten, Hastie, Tibshirani. Introduction to Statistical Learning with R.