EST-25134: Aprendizaje Estadistico

Profesor: Alfredo Garbuno Iñigo — Primavera, 2023 — Remuestreo.
Objetivo: Estudiaremos estrategias para cuantificar el error de generalización utilizando la información disponible. Revisaremos conceptos separación de muestras, validación cruzada y Bootstrap.
Lectura recomendada: Capítulo 5 de [1] y Capítulo 4 de [2].

1. INTRODUCCIÓN

Estudiaremos métodos de `remuestreo`: validación cruzada y *bootstrap* como mecanismos computacionales para obtener información adicional sobre el modelo ajustado.

Queremos evaluar la capacidad predictiva de nuestro modelo en instancias que no habíamos visto antes. El error cuantificado en el conjunto de entrenamiento usualmente es `optimista` en el sentido de estar sujeto a predecir en instancias ya observadas.

Esto es relevante en instancias donde hay parámetros (`hiper-parámetros`) que no son ajustados en el procedimiento de entrenamiento. Por ejemplo, el grado de polinomio en una regresión polinomial o el número de vecinos en un algoritmos de `vecinos mas cercanos`.

Hasta ahora, hemos considerado $\mathcal{D}_n$ y $\mathcal{T}_m$ como la separación de datos en dos conjuntos *provenientes* de la misma población. Nos permiten cuantificar:

- El **error de entrenamiento** $\mathcal{L}(\mathcal{D}_n)$.
- El **error de prueba** o `generalización` $\mathcal{L}(\mathcal{T}_m|\mathcal{D}_n)$.

La idea es poder predecir nuevas observaciones con el mismo nivel de precisión que observamos en el conjunto de entrenamiento. ¿Qué garantías tenemos para poder confiar en un modelo entrenado? Por supuesto, estamos tratando bajo los supuestos de que nuestros datos son representativos de la población sobre la cual haremos predicciones. Estamos sujetos al problema de `sobre-ajuste`.

1.1. ¿Cuál es la mejor manera para estimar de manera confiable el error de generalización?

(LGN) Un conjunto de datos grande que permita cuantificar *bien* el error.

1.2. ¿Qué alternativas tenemos?

1. Utilizar métodos que hacen correcciones al sesgo por utilizar el conjunto de datos para generalizar.
2. Utilizar un conjunto `fuera de entrenamiento`.

2. UTILIZANDO UN CONJUNTO DE VALIDACIÓN

Utilizamos una partición de nuestro conjunto de datos: `datos de entrenamiento` y `datos de validación`.

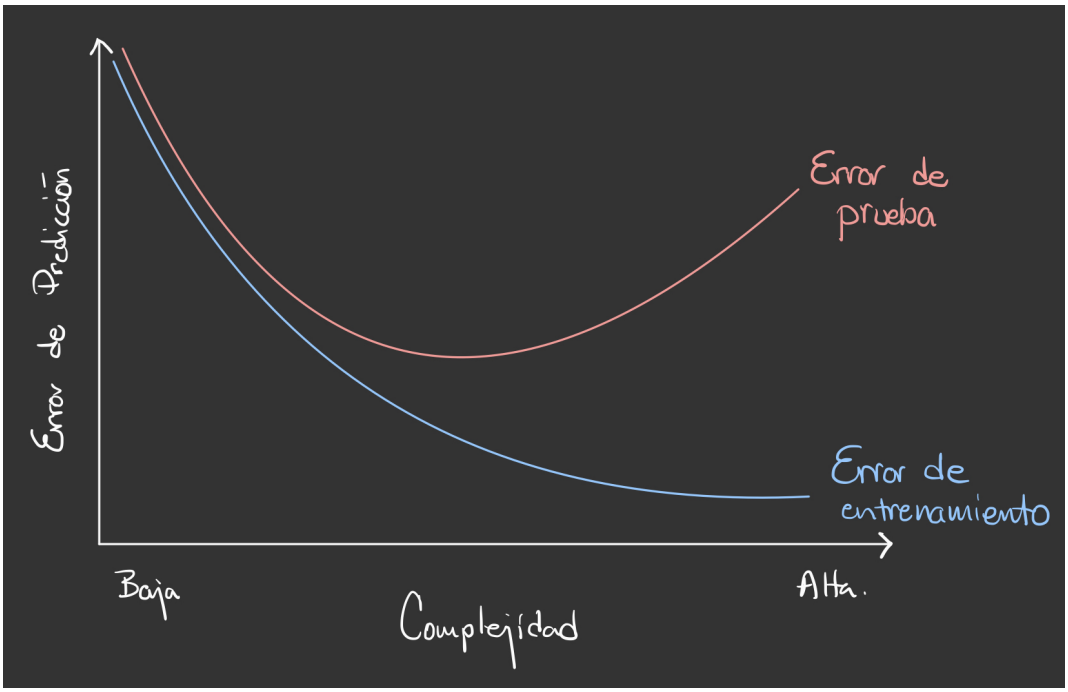

FIGURA 1. *Comportamiento típico del error de generalización y entrenamiento.*

El error cuantificado en el conjunto de validación nos da un estimador del error de prueba. Típicamente utilizamos el mismo criterio de error con el que entrenamos (RMSE, precisión, etc.).

2.1. Ejercicio: Autos

Obtener una predicción del rendimiento de combustible en términos de la potencia del motor.

```
## óSeparacin entrenamiento - prueba ----------------------------
library(rsample)

data ← read.csv("https://www.statlearning.com/s/Auto.csv") ▷
  as_tibble() ▷
  mutate(horsepower = as.numeric(horsepower)) ▷
  select(-name) ▷
  filter(!is.na(horsepower))
```

```
# A tibble: 6 × 8
    mpg cylinders displacement horsepower weight acceleration  year origin
  <dbl>     <int>        <dbl>      <dbl>  <int>        <dbl> <int>  <int>
1    18         8          307        130   3504         12      70      1
2    15         8          350        165   3693         11.5    70      1
3    18         8          318        150   3436         11      70      1
4    16         8          304        150   3433         12      70      1
5    17         8          302        140   3449         10.5    70      1
6    15         8          429        198   4341         10      70      1
```

Dividimos nuestro conjunto en `entrenamiento` y `prueba`.

```
set.seed(108790)
sample_rows ← sample(1:nrow(data), nrow(data)/2)

data_train ← data[sample_rows,]
data_test ← data[-sample_rows,]
```

Listing 1. *Separación de datos en entrenamiento validación.*

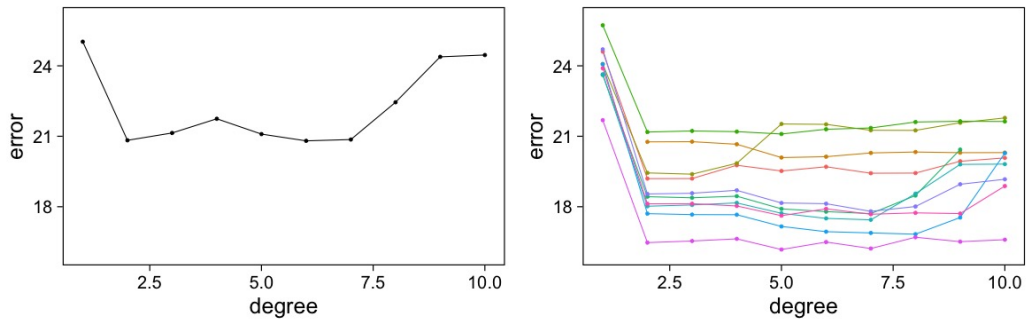

Figura 2. *Error de validación utilizando* una *partición del conjunto de datos (izquierda). Error de validación con distintas particiones de los datos (derecha).*

Nota que las estimaciones son muy variables. Son altamente sensibles al conjunto de entrenamiento y validación que se utillizaron. Podríamos estar **sobre-estimando** el error de generalización pues dejamos de lado un conjunto de datos para entrenar el modelo.

3. VALIDACIÓN CRUZADA

Es una técnica que nos permite elegir la *mejor* configuración de un modelo y nos da indicios del error de *generalización*. La idea es:

1. Dividir el conjunto de datos en K bloques.
2. Utilizar un método iterativo para ajustar modelos con $K-1$ bloques y registrar el error de ajuste con el bloque fuera del entrenamiento.

3.1. Detalles (regresión)

Sean K bloques y utilicemos $C_1, C_2, \ldots C_K$ para denotar con C_k el conjunto de índices en el bloque k. En total tenemos n_k observaciones en cada bloque. Un caso particular es $n_k = n/K$.

En cada iteración (k) `calculamos el error de predicción` (MSE_k) sobre el `conjunto que dejamos fuera del entrenamiento`. Promediamos todos los errores para reportar el error de pérdida bajo validación cruzada.

$$\mathsf{CV}_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} \mathsf{MSE}_k \,. \tag{1}$$

3.2. Caso especial: `LOO-CV`

Si utilizamos $K = n$, entonces tenemos lo que se conoce como *leave-one out cross-validation* (`LOO-CV`).

En el `caso de estimadores lineales` por mínimos cuadrados (como regresión polinomial) tenemos un *atajo* para calcular con *un sólo ajuste*

$$\mathsf{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1-h_i}\right)^2 . \tag{2}$$

Donde h_i es el estadístico de `anclaje` de la observación i.

La definición de este estadístico lo puedes encontrar en el Capítulo 3 de [1] en la página 99,

$$h_i = \frac{1}{n} + \frac{(x_i - \bar{x}_n)^2}{\sum_{j=1}^{n}(x_j - \bar{x}_n)^2} . \tag{3}$$

3.3. Pseudocódigo - validación cruzada

Podemos usar las funciones de la librería `rsample`.

```
## óValidacin cruzada ----------------------------------
ajusta_modelo ← function(split){
    ## Separa en entrenamiento / validacion
    train ←  analysis(split)
    valid ← assessment(split)
    ## Entrena y evalua
    tibble(degree = 1:10) ▷
      mutate(model = map(degree, fit_model, train),
             error = map_dbl(model, eval_error, valid))
  }
```

Listing 2. *Código ejemplo para procesar datos en entrenamiento y validación.*

```
data ▷ vfold_cv(5)
data ▷ loo_cv()
```

Listing 3. *Funciones para hacer los bloques de validación cruzada.*

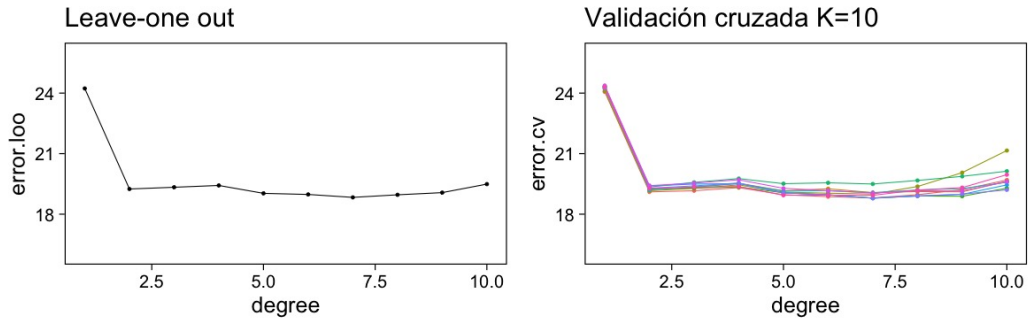

Figura 3. *Métricas de error bajo validación cruzada.*

3.4. Observaciones

Utilizamos conjuntos de datos mas pequeños para entrenar. Por lo tanto tenemos un sesgo en el error mas grande de lo que hubiéramos querido.

El sesgo se *puede eliminar* al tomar $K = n$ pero tiene una *gran varianza*.

> Al tener bloques de entrenamiento de tamaño $n - 1$ con una alta probabilidad habrá correlación en los MSE_k lo que ocasiona que se infle la varianza.

En la práctica un *buen compromiso* se puede establecer con $K = 5$ ó 10 (experimentación empírica).

3.5. ¿Y para clasificación?

3.6. Un caso para pensar

Consideremos que tenemos un conjunto de datos con pocas muestras y muchos atributos, $p \gg n$. Para ajustar un modelo lo que hacemos es:

1. Encontrar los $p = 20$ predictores con mayor correlación con la respuesta.
2. Utilizar validación cruzada para entrenar un modelo con esos $p = 20$ predictores y cuantificar su error de generalización.

¿Está bien esta estrategia?

3.7. ¿Cómo escoger K?

La elección usual es 5 ó 10 (en principio cualquier elección en este intervalo). Lo que queremos es poder estimar el error de generalización. Sin embargo, el estimador de error por validación cruzada puede tener tanto `sesgo` o `varianza` elevada.

Como mencionamos antes, con `LOO-CV` tenemos bloques altamente correlacionados lo cual nos contamina la estimación de varianza (error estándar) de nuestro estimador aunque con un sesgo mas pequeño.

Validación cruzada con un número limitado de bloques nos puede ayudar a controlar la varianza (¿por qué?) aunque a un costo de aumento en sesgo.

En la práctica, podemos hacer varias réplicas del procedimiento de validación cruzada (utilizando distintas particiones en K bloques) para mejorar nuestras estimaciones del error estándar y *mejorar* nuestra cuantificación del valor esperado del estimador.

Nota: Por *mejorar* no hacemos referencia a disminuir la incertidumbre (error estándar o amplitud de un intervalo) si no a una estimación mas cercana a los valores reales.

4. BOOTSTRAP

Es una técnica de remuestreo que nos permite cuantificar incertidumbre sobre un *estimador* o un *procedimiento de estimación*.

Lo usamos muchas veces para estimar el *error estándar* de un estimador o poder reportar intervalos de confianza basados en percentiles. (No utilizamos supuestos asintóticos).

Si pudiéramos generar muestras de la población no tendríamos problemas. Pero en muchas ocasiones no tenemos acceso al generador de datos.

Resolvemos estos problemas tomando `re-muestras` de las observaciones que tenemos utilizando `muestreo aleatorio con reemplazo`.

De esta manera, creamos conjuntos de datos ficticios (a partir de los datos observados) que nos permiten estimar las cantidades de interés. Con un número suficiente de réplicas podemos obtener una distribución de estimadores de la cual podemos extraer percentiles para construir un intervalo de confianza.

4.1. Observaciones

Muchas veces hay que tener cuidado con la forma en que generamos las remuestras. Por ejemplo, en situaciones con datos temporales o geográficos.

4.2. Cuantificando el error de generalización

En validación cruzada los bloques no tienen traslape. Esto es ventajoso para cuantificar el error y su variación.

Si utilizáramos *bootstrap* entonces los bloques ocasionarían problemas con los estimadores. Esto es por que aproximadamente el 63 % de las observaciones se repiten en el muestreo con reemplazo. Esto es equivalente a una validación cruzada con $K \approx 2$ bloques.

Utilizar *bootstrap* implica utilizar un mecanismo de muestreo aleatorio **con** reeemplazo. Si tenemos una colección de n instancias y queremos calcular la probabilidad de escoger **al menos una vez** la instancia i -ésima, lo calculamos por medio de

$$1 - \mathbb{P}(\text{no escoger el índice } i) = 1 - \left(1 - \frac{1}{n}\right)^n \approx 1 - e^{-1} \approx 63.2\,\%\,. \tag{4}$$

REFERENCIAS

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer Texts in Statistics. Springer US, New York, NY, 2021.

[2] M. Kuhn and K. Johnson. *Applied Predictive Modeling*. Springer New York, New York, NY, 2013. ISBN 978-1-4614-6848-6 978-1-4614-6849-3. .