Remuestreo

1EST17 - Aprendizaje Estadístico I

Enver Tarazona

2022-10-14

Introducción

Material de aprendizaje para esta sesión

- James et al (2021): An Introduction to Statistical Learning. Chapter 5.
- Material adicional de interés (nivel más avanzado): Capítulo 7 (en particular 7.10) en Friedman et al (2001): Elements of Statistical learning.

¿Qué aprenderemos?

- ¿Cómo se evalúa y selecciona un modelo predictivo?
- Solución ideal en una situación de abundancia de datos.
- Validación Cruzada (CV): Conjunto de validación - LOOCV y k-fold CV - ¿Cuál es mejor?
- Bootstrapping - Como y por qué.

Eficiencia de un método de aprendizaje.

- Nuestros modelos son "buenos" cuando pueden generalizar.
- Queremos un método de aprendizaje que funcione bien con datos nuevos (error de prueba bajo).
- Inferencia y comprensión del patrón verdadero (en contraste con el sobreajuste)

Esto es importante para:

Selección de Modelos

Estimar el *rendimiento* predictivo de diferentes modelos (generalmente diferente orden de complejidad de un mismo tipo modelo) para *elegir el mejor*.

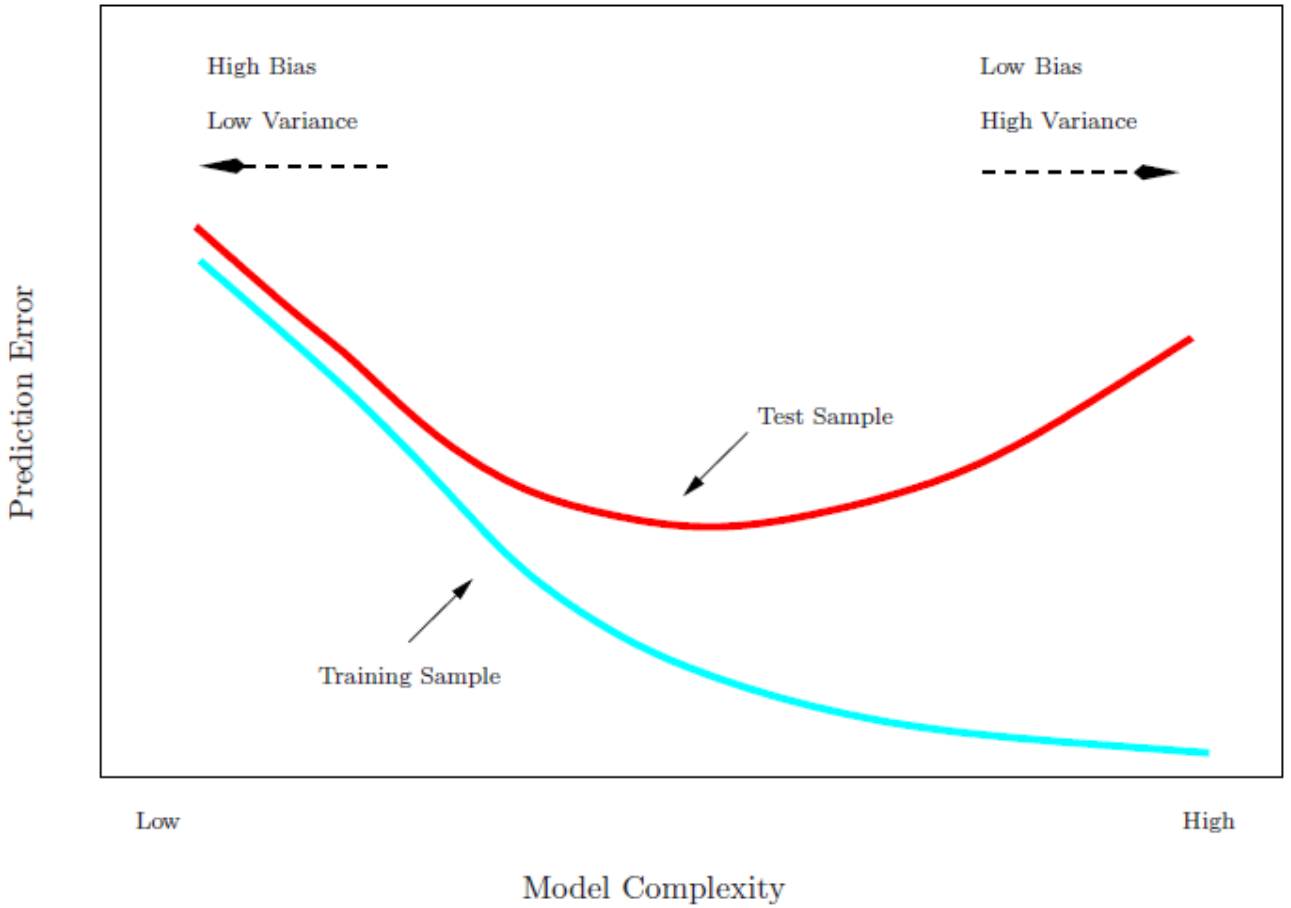
Evaluación de Modelos

Estimar su rendimiento (error de predicción) del modelo final sobre un nuevo conjunto de datos.

Error de entrenamiento vs. error de prueba

Recordar:

- El *error de prueba* es el error promedio que resulta de usar un método de aprendizaje estadístico para predecir la respuesta en una nueva observación, una que no se usó para entrenar el método.
- El *error de entrenamiento* se puede calcular fácilmente aplicando el método de aprendizaje estadístico a las observaciones utilizadas en su entrenamiento.
- La tasa de error de entrenamiento a menudo es bastante diferente de la tasa de error de prueba.
- **El error de entrenamiento puede subestimar drásticamente el error de prueba**.


High Bias
Low Variance
Low Bias
High Variance
Test Sample
Training Sample
Prediction Error
Low
High
Model Complexity

Funciones de Pérdida

Se suele usar:

- *Error Cuadrático Medio* (pérdida cuadrática) para problemas de regresión (respuesta continua) $Y_i = f(x_i) + \varepsilon_i$, $i = 1, \ldots, n$:

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

  De forma equivalente se usa RMSE (raíz del error cuadrático medio)

- *Ratio de Mala Clasificación* (pérdida 0/1) para problemas de clasificación:

$$P(Y = j \mid \mathbf{x}_0) \text{ para } j = 1, \ldots, K$$

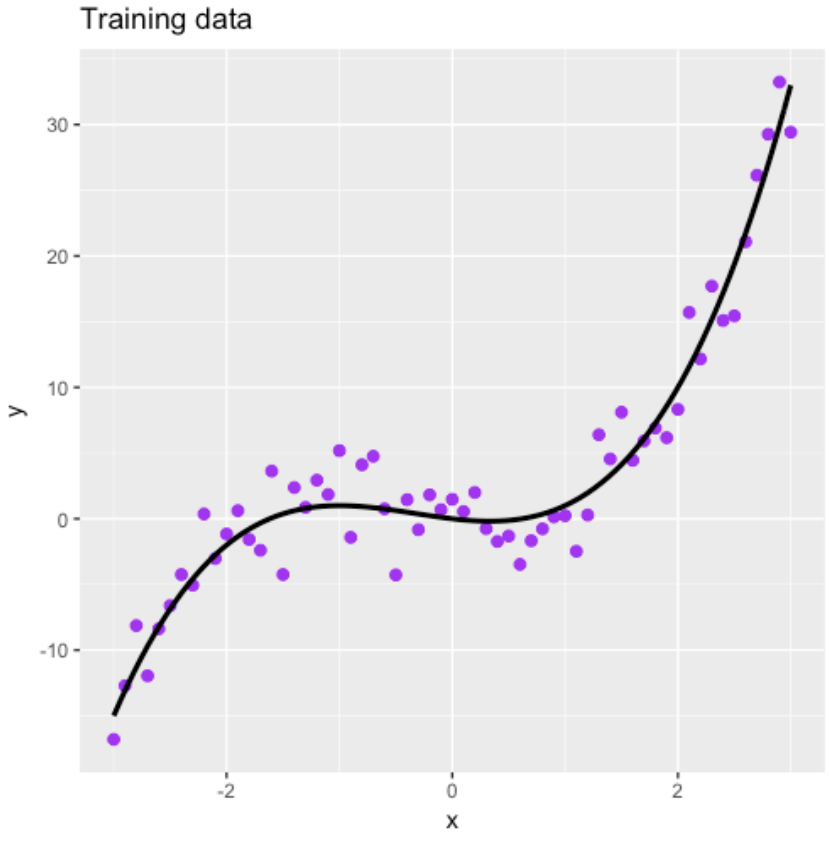
  y clasificar a la clase con mayor probabilidad $\hat{y}_i$. Entonces

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

  nos dará el ratio total del error de clasificación.

Ejemplo

Nuestro objetivo es efectuar *selección de modelos* en regresión KNN (K vecinos más cercanos), donde la curva verdadera es $f(x) = -x + x^2 + x^3$ con $x \in [-3, 3]$. $n = 61$ para los datos de entrenamiento.



Regresión KNN (capítulo 3.5 del libro ISLR)

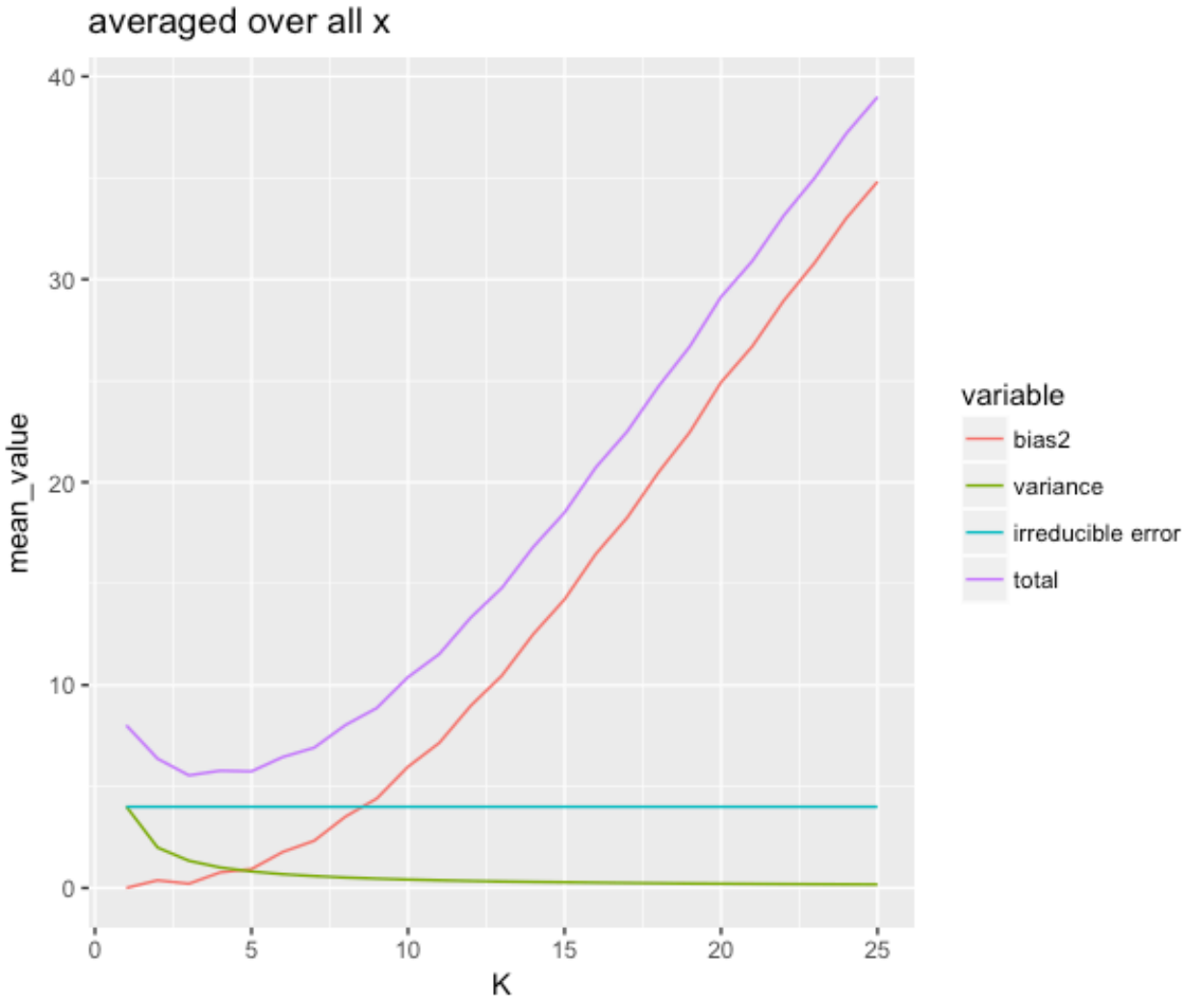
- El método de regresión KNN proporciona una predicción a un valor x_0 encontrando los puntos K más cercanos (distancia euclidiana) y calculando el promedio de los valores y observados en los puntos en la vecindad respectiva $\mathcal{N}_0$

$$\hat{f}(x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} y_i \ .$$

- Hemos considerado $K = 1, \ldots, 25$, y repetimos el experimento $M = 1000$ veces (es decir, M versiones conjunto de entrenamiento y prueba).

Recordar: el balance sesgo-varianza

Para KNN: K pequeño = alta complejidad; K grande = menor complejidad.



El reto

- En los ejemplos anteriores sabíamos la verdad, por lo que pudimos evaluar el error de entrenamiento y prueba.
- En realidad, este no es el caso, por supuesto.
- ¡Necesitamos enfoques que funcionen con datos reales!

Situación con abundancia de datos (ideal pero usualmente no realista)

Si tenemos suficiente cantidad de datos podemos dividir nuestro conjunto de datos en tres partes:

- **Conjunto de entrenamiento** (training set): para ajustar (estimar) el modelo
- **Conjunto de validación** (validation set): para seleccionar el mejor modelo (*selección de modelos*)
- **Conjunto de evaluación** o prueba (test set): para evaluar que tan bien el modelo se ajusta a un conjunto nuevo e independiente de datos (*evaluación de modelos*)

P: Antes solo teníamos conjuntos de datos de entrenamiento y prueba. ¿Por qué necesitamos el conjunto de validación adicional?

R: No habíamos hablado antes de la selección del modelo.

P: ¿Por qué no podemos usar un conjunto de datos para entrenamiento, y otro conjunto de datos para seleccionar el modelo y también evaluarlo?

R: Porque podríamos estar siendo demasiado optimistas si reportamos el error obtenido sobre el conjunto de evaluación cuando ya lo hemos usado para seleccionar el mejor modelo.

- Si estamos en esta situación - ¡excelente! - entonces no necesitamos continuar con el resto de este tema.
- Pero, esto suele ser un caso muy raro - por lo que se estudiarán otras soluciones basadas en un uso eficiente de la muestra aplicado técnicas de *remuestreo* de datos.
- Una estrategia alternativa para seleccionar el modelo es usando métodos de penalización por complejidad, p.j. AIC o lasso.

Primero hablaremos de *validación cruzada*, y luego de *bootstrap*.

Validación Cruzada (CV)

Situación de "selección de modelos": Vamos a asumir que el conjunto de datos de evaluación se encuentra disponible (y ha sido separado), y queremos usar el resto de nuestros datos para encontrar el modelo que se desempeña "mejor", es decir, *con el error de prueba más bajo*.

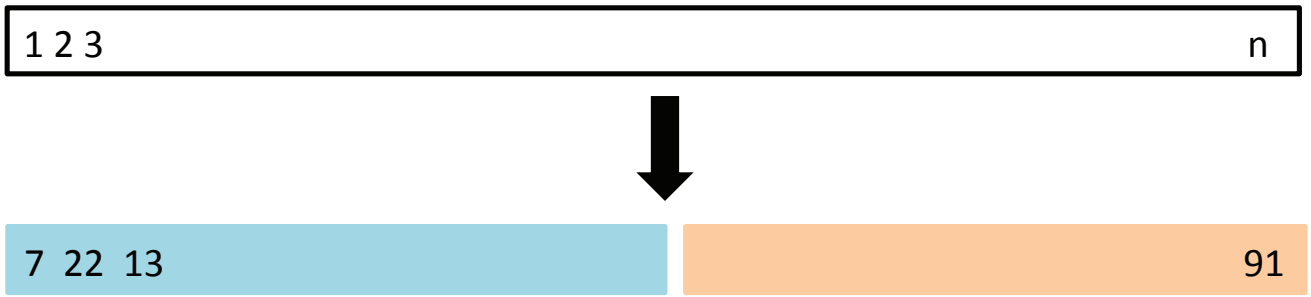
Esto puede ser realizado a través de:

- El esquema de un conjunto de validación o método de retención (estrictamente hablando no es validación cruzada)
- Validación cruzada dejando uno afuera (LOOCV)
- Validación cruzada con K iteraciones (*K-fold Cross Validation*), usualmente en 5 y 10 grupos (K-CV)

Enfoque usando un conjunto de validación

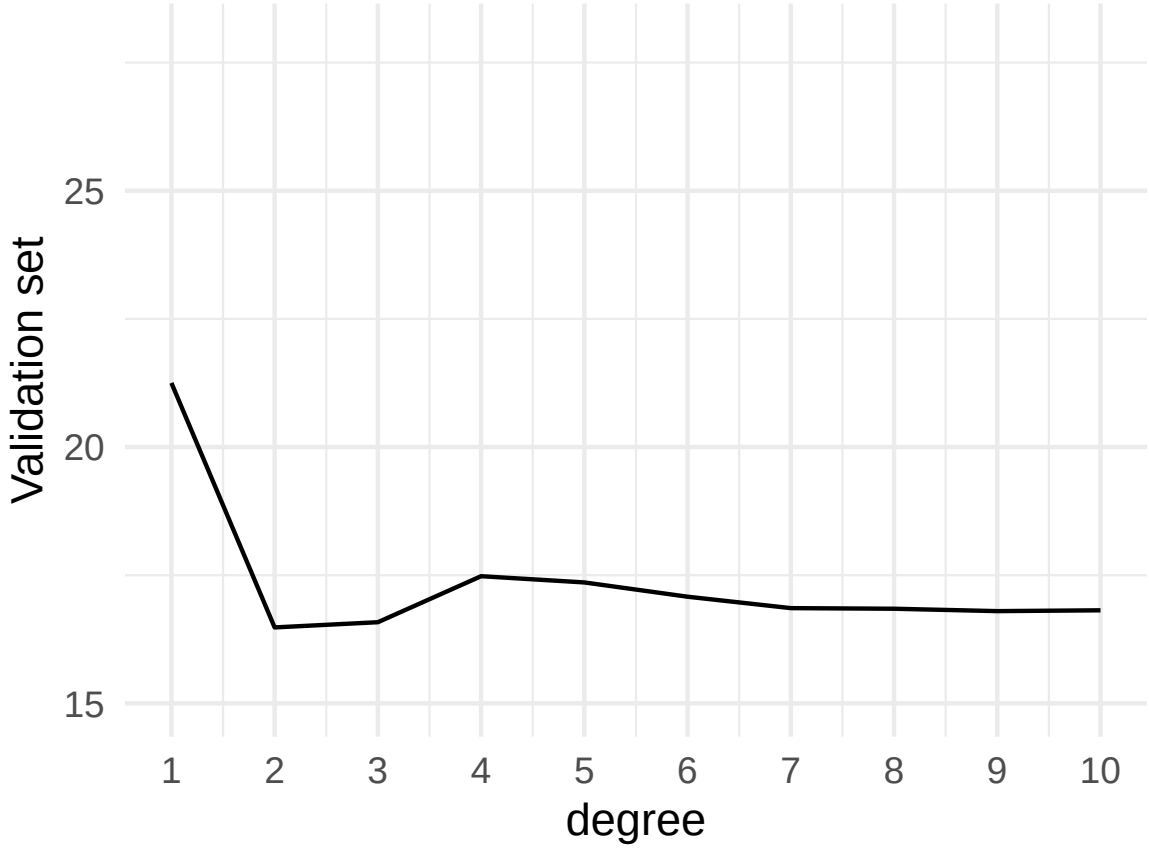
- Consideremos el caso cuando se dispone de un conjunto de datos con n observaciones.
- Para ajustar el modelo y evaluar su rendimiento predictivo se divide aleatoriamente el conjunto de datos en dos partes (tamaño de muestra para cada parte $n/2$):
  - Un *conjunto de entrenamiento* (para ajustar el modelo) y
  - Un *conjunto de validación* (para realizar predicciones de la variable respuesta para cada observación en el conjunto de validación)

Recordar: Enfoque en seleccionar el mejor modelo.

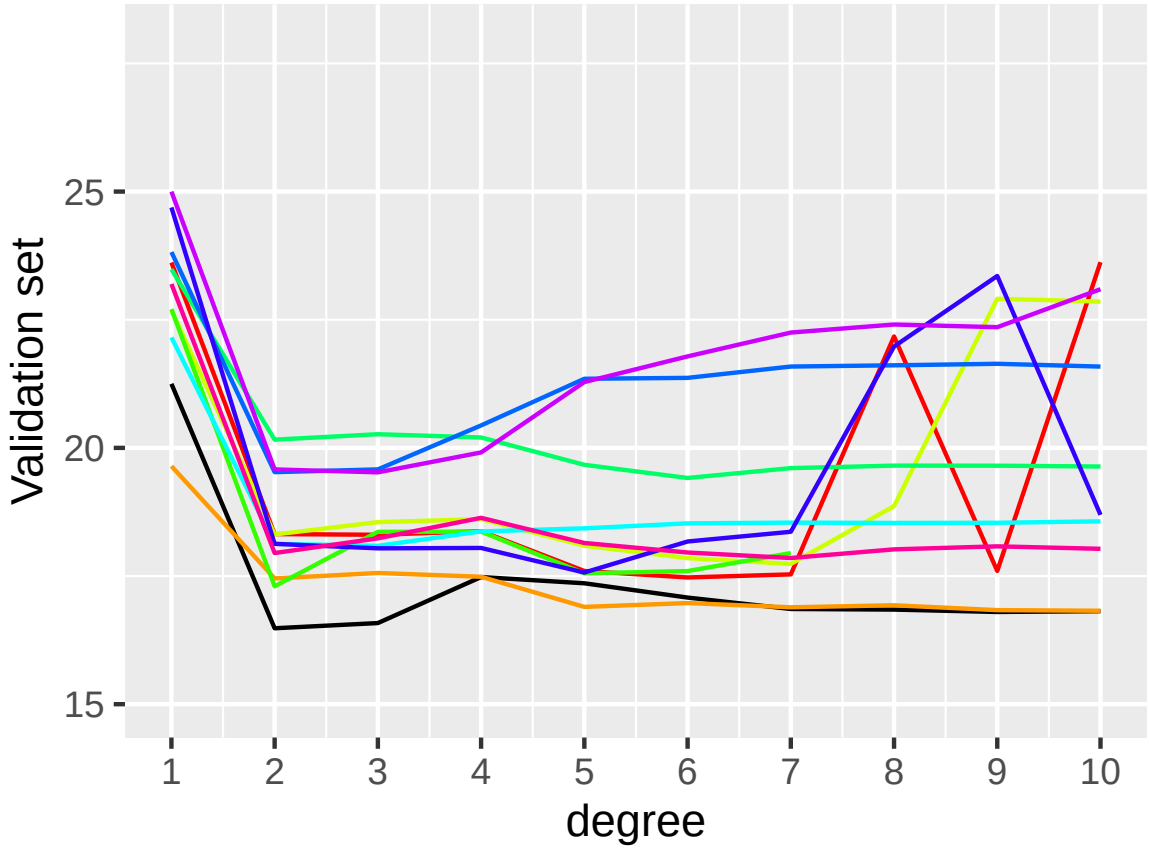


Ejemplo: error en un conjunto de validación

Conjunto de datos Auto (library `ISLR`): predecir `mpg` (millas por galón) usando una función polinomial de `horsepower` (caballos de fuerza del motor), $n = 392$. ¿Qué es lo que se observa?



Ejemplo: error en un conjunto de validación para varias selecciones



Inconvenientes del enfoque del conjunto de validación

- *Alta variabilidad* del error en el conjunto de validación - dado de que es dependiente de que observaciones son incluidas en el conjunto de entrenamiento y validación (problema de varianza).

- *Tamaño de muestra más pequeño* para el ajuste del modelo, ya que solo la mitad de las observaciones están en el conjunto de entrenamiento. Por lo tanto, el error del conjunto de validación puede tender a sobreestimar la tasa de error en nuevas observaciones para un modelo que se ajusta al conjunto de datos completo (problema de sesgo).

Validación cruzada dejando uno afuera (LOOCV)

La validación cruzada de dejar uno fuera (LOOCV) aborda las limitaciones del enfoque del conjunto de validación.

Idea:

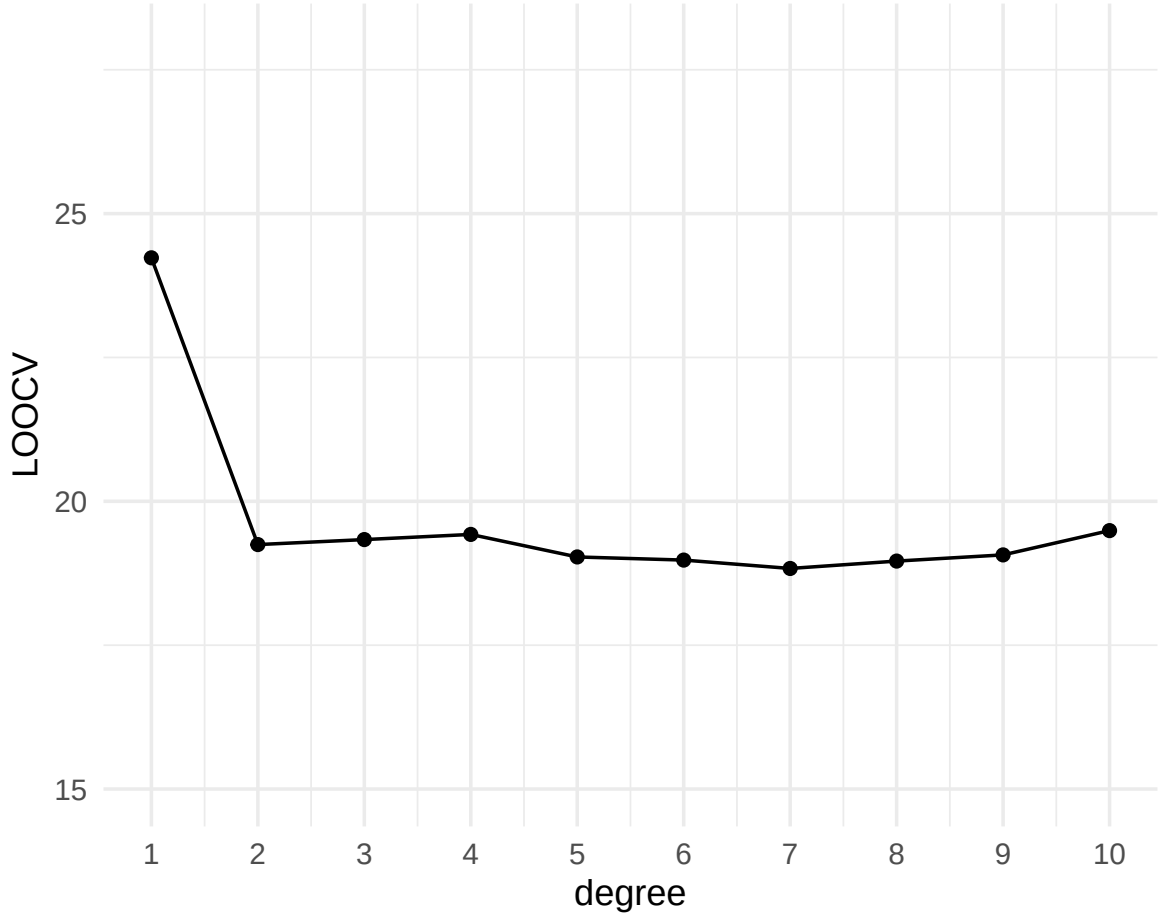
- Solo **Una observación a la vez** se omite y se considera como nueva observación (conjunto de prueba).
- Las observaciones $n-1$ restantes constituyen el conjunto de entrenamiento.
- El procedimiento de ajuste del modelo se repite n veces, de modo que cada una de las n observaciones es dejada afuera una vez. En cada paso, calculamos el MSE como

$$\text{MSE}_i = (y_i - \hat{y}_i)^2$$

- El **error total de predicción** es la media obtenida con los n modelos

$$CV_n = \frac{1}{n}\sum_{i=1}^{n} \text{MSE}_i$$

Ejemplo de Regresión: LOOCV



```r
library(ISLR)  #Para cargar el conjunto de datos Auto
library(boot)  #Para cv.glm
library(ggplot2)  #Para graficar
set.seed(123)
n = dim(Auto)[1]
testMSEvec = NULL
start = Sys.time()
for (polydeg in 1:10) {
    glm.fit = glm(mpg ~ poly(horsepower, polydeg), data = Auto)
    glm.cv1 = cv.glm(Auto, glm.fit, K = n)
    testMSEvec = c(testMSEvec, glm.cv1$delta[1])
}
stopp = Sys.time()
yrange = c(15, 28)
plotdf = data.frame(testMSE = testMSEvec, degree = 1:10)
g0 = ggplot(plotdf, aes(x = degree, y = testMSE)) + geom_line() + geom_point() +
    scale_y_continuous(limits = yrange) + scale_x_continuous(breaks = 1:10) +
    labs(y = "LOOCV")
g0 + theme_minimal()
```

Ventajas y desventajas de LOOCV

- Pros:
  - ¡Sin aleatoriedad en las divisiones de entrenamiento/validación!
  - Poco sesgo, ya que casi todo el conjunto de datos se utiliza para el entrenamiento (en comparación con la mitad para el enfoque del conjunto de validación).
- Contras:
  - Implementación costosa - se requiere ajustar n modelos diferentes - sin embargo, puede ser un buen enfoque para un modelo lineal LOOCV - pero generalmente no es así.
  - Alta varianza: Dos conjuntos de datos solo difieren en una observación - esto hace que las estimaciones de cada parte tengan una alta correlación y esto puede conducir a que el promedio tenga una alta varianza. Recordar:

$$\text{Var}(\sum_{i=1}^{n} a_i X_i) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j \text{Cov}(X_i, X_j)$$

$$= \sum_{i=1}^{n} a_i^2 \text{Var}(X_i) + 2\sum_{i=2}^{n}\sum_{j=1}^{i-1} a_i a_j \text{Cov}(X_i, X_j).$$

LOOCV en Regresión Lineal Múltiple

Hay algo bueno para LOOCV en el caso de regresión lineal:

$$\mathrm{CV}_n = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_{ii}}\right)^2 \quad ,$$

donde h_i esl el ith elemento de la diagonal de la matriz hat (leverage) $\mathbf{H} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ y $\hat{y}_i$ es el i-ésimo valor ajustado por mínimos cuadrados del modelo original..

$\rightarrow$ ¡Necesita ajustar el modelo solo una vez!

Validación Cruzada en k-grupos (k-fold CV)

Para abordar los inconvenientes de LOOCV, podemos omitir no solo una única observación en cada iteración, sino la $1/k$-ésima parte de todos los datos.

Procedimiento:

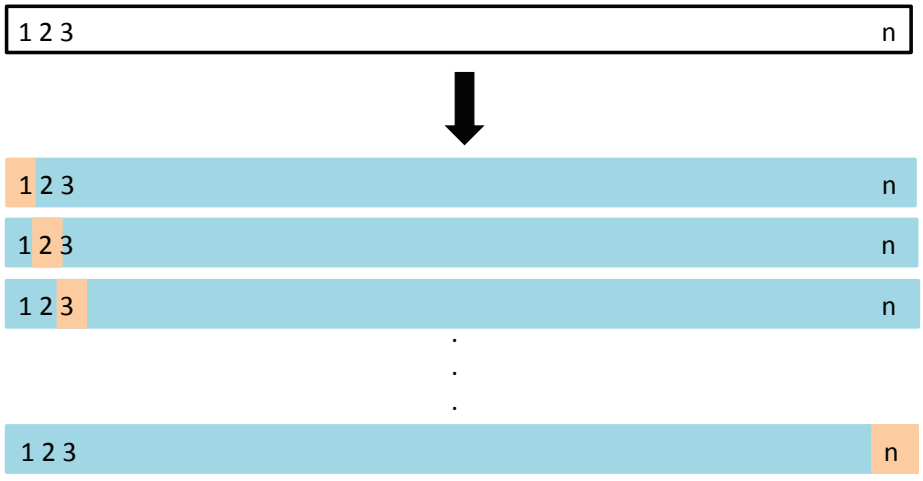
- Divida los datos en k partes (más o menos) iguales.
- Utilice $k-1$ partes para ajustar y la k ésima parte para validar.
- Haga esto k veces y omita otra parte en cada ronda.

El MSE es entonces estimado en cada una de las iteraciones de k ($\text{MSE}_1, \ldots, \text{MSE}_k$), y el MSE por CV en k-grupos es

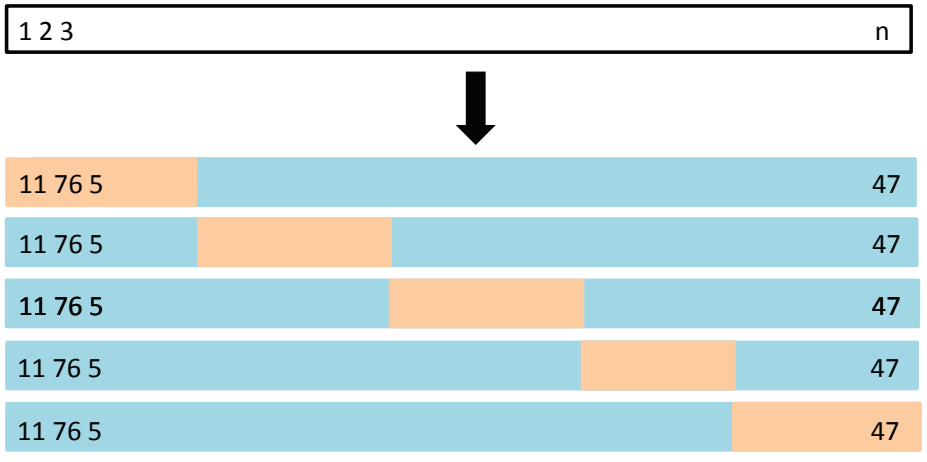
$$\text{CV}_k = \frac{1}{k} \sum_{i=1}^{k} \text{MSE}_i \ .$$

Comparación entre LOOCV y k-fold CV:

LOOCV:



k-fold:



Formalmente

- Índices de observaciones - Dividir en k grupos (folds): $C_1, C_2, \ldots, C_k$.
- n_k elementos en cada grupo, si n es un múltiplo de k entonces $n_k = n/k$.

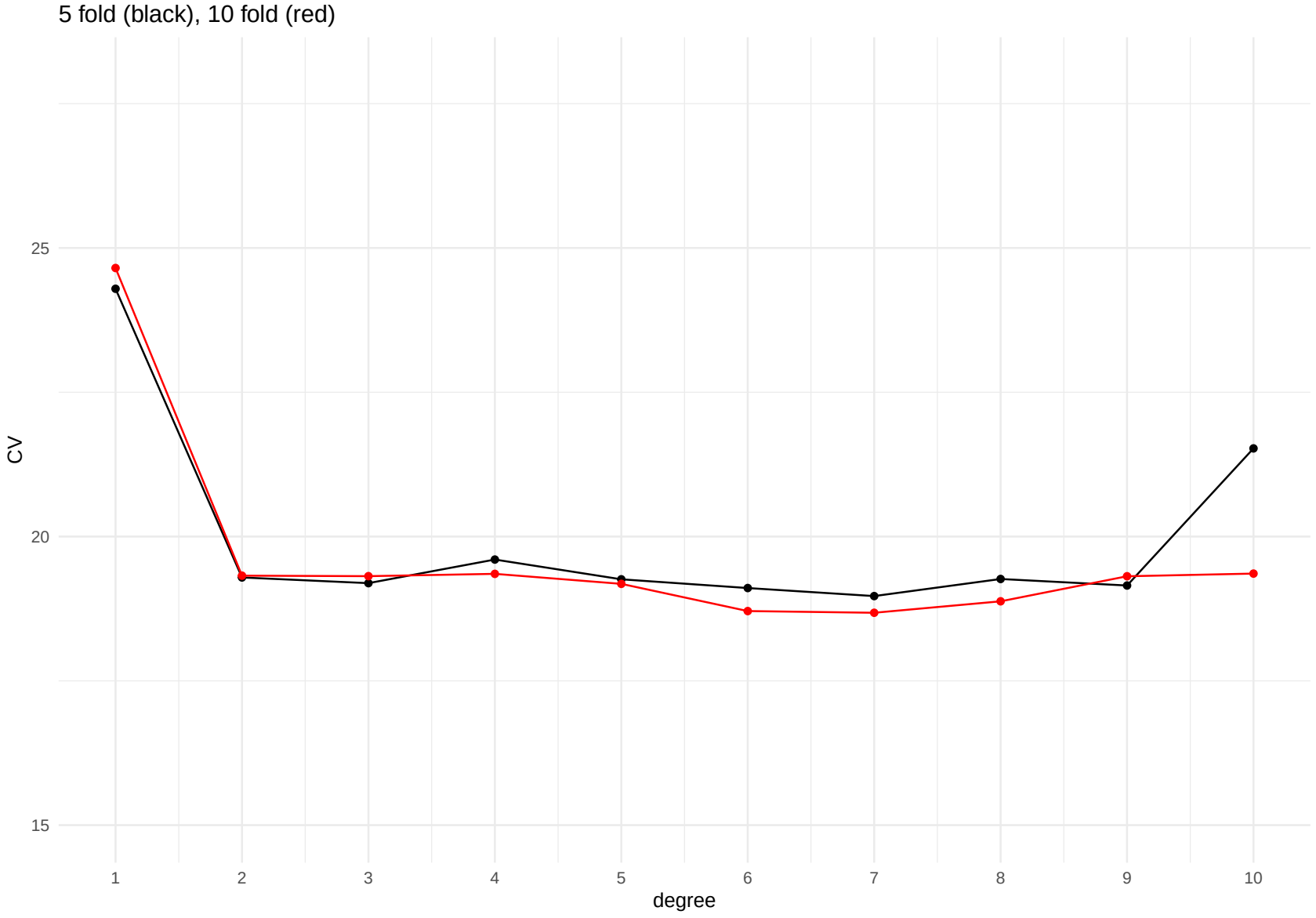
$$\text{MSE}_k = \frac{1}{n_k} \sum_{i \in C_k} (y_i - \hat{y}_i)^2$$

donde $\hat{y}_i$ es el valor ajustado para la observación i obtenida de los datos de la partición k removida.

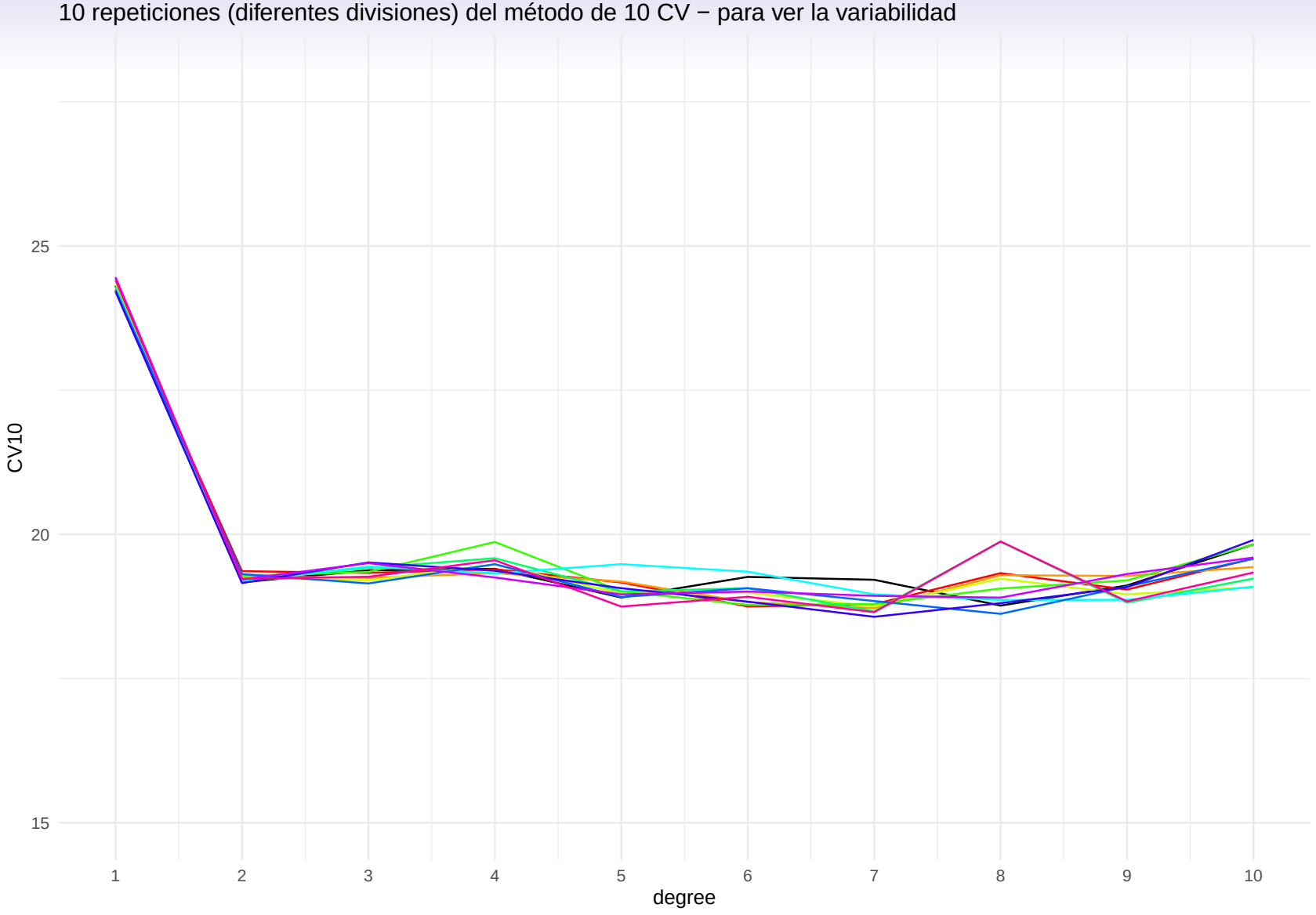
$$\text{CV}_k = \frac{1}{n} \sum_{j=1}^{k} n_j \text{MSE}_j$$

Observar: Definir $k = n$ da LOOCV.

Ejemplo: Validación cruzada con 5 y 10-grupos



```
library(ISLR)
library(boot)
library(ggplot2)
set.seed(123)
n = dim(Auto)[1]
testMSEvec5 = NULL
testMSEvec10 = NULL
start = Sys.time()
for (polydeg in 1:10) {
    glm.fit = glm(mpg ~ poly(horsepower, polydeg), data = Auto)
    glm.cv5 = cv.glm(Auto, glm.fit, K = 5)
    glm.cv10 = cv.glm(Auto, glm.fit, K = 10)
    testMSEvec5 = c(testMSEvec5, glm.cv5$delta[1])
    testMSEvec10 = c(testMSEvec10, glm.cv10$delta[1])
}
stopp = Sys.time()
yrange = c(15, 28)
plotdf = data.frame(testMSE5 = testMSEvec5, degree = 1:10)
g0 = ggplot(plotdf, aes(x = degree, y = testMSE5)) + geom_line() + geom_point() +
    scale_y_continuous(limits = yrange) + scale_x_continuous(breaks = 1:10) +
    labs(y = "CV") + ggtitle("5 and 10 fold CV")
g0 + geom_line(aes(y = testMSEvec10), colour = "red") + geom_point(aes(y = testMSEvec10),
    colour = "red") + ggtitle("5 fold (black), 10 fold (red)") + theme_minimal()
```



Todavía hay *variabilidad*, pero *mucho menos* que para el enfoque del conjunto de validación.

Ventajas y desventajas con k-fold cross-validation

1. Al igual que el esquema del conjunto de validación, el resultado puede variar según la forma en que se seleccionan los grupos, pero la variación es en general más baja que para el enfoque del conjunto de validación.
2. Aspecto computacional: menos trabajo con $k = 5$ o 10 que con LOOCV.
3. El sesgo es menor cuando $k = n$ (LOOCV), pero sabemos que LOOCV tiene una alta varianza.
4. Debido al *balance entre sesgo y varianza*, k-fold CV a menudo proporciona estimaciones más precisas de la tasa de error de prueba que LOOCV. $\rightarrow$ Esto es por lo que usualmente $k = 5$ o $k = 10$ es usado.

Elección del mejor modelo

Recordar que se *divide aleatoriamente* los datos en k grupos y luego se realiza CV for para todos los posibles modelos que queremos comparar.

- Hay un hiperparámetro del modelo (tal vez K en KNN o el grado del polinomio), digamos θ, involucrado para calcular CV_j, $j = 1, \ldots, k$
- Basado en la gráfica del CV vs θ, podemos elegir el modelo con *el menor* CV_k como nuestro mejor modelo.
- Luego ajustamos este modelo utilizando todo el conjunto de datos (no la parte de prueba, que aún se mantiene alejada) y evaluamos el rendimiento en el conjunto de prueba.

Regla del error estándar:

Denotemos por $\text{MSE}_j(\theta)$, $j = 1, \ldots, k$ las k partes del MSE que en conjunton dan CV_k.

Podemos calcular la desviación estándar muestral (error estándar) de todos los $\text{MSE}_j(\theta)$, $j = 1, \ldots, k$

$$\hat{\text{SE}}(\text{CV}_k(\theta)) = \sqrt{\sum_{j=1}^{k} (\text{MSE}_j(\theta) - \overline{\text{MSE}}(\theta))/(k-1)}$$

para cada valor del parámetro de complejidad θ.

Estrictamente esto no es válido y deberíamos usar $\text{SE}(\theta) = \text{SD}(\theta)/\sqrt{k}$ como la desviación estándar de CV_k.

Regla del error estándar:

La *regla del error estándar* consiste en elegir el modelo más simple (e.g. mayor valor de K en KNN o menor grado de un polinomio) θ, tal que, conducimos a θ en la dirección de simplicidad hasta que sea verdadero que

$$\mathrm{CV}(\theta) \leq \mathrm{CV}(\hat{\theta}) + \hat{\mathrm{SE}}(\mathrm{CV}_k(\hat{\theta})) \ .$$

Este será el modelo más simple cuyo error se encuentra a una desviación estándar del mínimo error.

k-fold cross-validation en clasificación

- ¿Qué se requiere cambiar en relación a la regresión?

Para LOOCV $\hat{y}_i$ es el ajuste de la observación i, obtenida de los datos con la observación i eliminada, y $\text{Err}_i = I(y_i \neq \hat{y}_i)$. LOOCV será entonces

$$CV_n = \frac{1}{n} \sum_{i=1}^{n} \text{Err}_i$$

Para k-fold CV se define de forma similar.

¿Podemos usar CV para evaluación de modelos ?

- Asuma que tenemos un método en el cuál no se requiere realizara una selección de modelos (por ejemplo, esto podría usarse con métodos de comparación de modelos usando AIC o lasso), pero si queremos evaluar el modelo basados en todos nuestros datos.
- Entonces podemos usar CV con todos los datos (entonces el conjunto de validación es realmente el conjunto de prueba) y reportar el rendimiento del modelo usando los conjuntos de prueba.

¿Podemos usar CV para seleccionar y evaluar un modelo de forma simultánea?

- Realmente no: el uso del conjunto de pruebas tanto para la selección del modelo como para la estimación tiende a sobreajustarse a los datos de prueba, y el sesgo se subestima.
- Solución: Se debe de aplicar CV en dos capas - también llamado *CV anidado*.
- Bucle externo: Evaluación del modelo (5-CV)
- Bucle interno: Selección del modelo (10-CV)

Bootstrap

- Herramienta estadística flexible y potente que se puede utilizar para cuantificar la *incertidumbre* asociada con un estimador o método de aprendizaje estadístico.

- Muy popular para obtener errores estándar o intervalos de confianza para un coeficiente, cuando la teoría paramétrica no lo proporciona.

- Veremos cómo obtener una estimación del error estándar de una mediana muestral y de un coeficiente de regresión.

- El inventor: Bradley Efron en 1979 - ver entrevista.
- ¿El nombre? *To pull oneself up by one's bootstraps* de "Las sorprendentes aventuras del barón Munchausen" de Rudolph Erich Raspe:

The Baron had fallen to the bottom of a deep lake. Just when it looked like all was lost, he thought to pick himself up by his own bootstraps.

- **Idea: Utilice los datos en sí para obtener más información sobre una estadística (un estimador).**

Ejemplo: La media $\bar{X}$

- Supongamos que tenemos variables aleatorias univariadas $X_1, X_2, \ldots, X_n$, con una media μ y varianza σ^2 común.
- Entonces podemos formar el promedio: $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$, el que usamos a menudo como estimador de μ.
- Si queremos estimar σ^2, un popular estimador es $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2$,
- Si las observaciones son *independientes* (e idénticamente distribuidas) conocemos que la varianza de $\bar{X}$ está dada por $\mathrm{Var}(\bar{X}) = \frac{\sigma^2}{n}$, la cuál puede ser estimada por $\widehat{\mathrm{Var}}(\bar{X}) = \frac{S^2}{n}$.
- Para formar un intervalo de confianza para μ necesitamos asumir una distribución para las X_is. Si nosotros asumimos que las X_is son independientes y normalmente distribuidas $N(\mu, \sigma^2)$ es posible obtener un intervalo del 95% de confianza a partir de:

$$\bar{X} \pm t_{0.025,n-1}\frac{S}{\sqrt{n}}$$

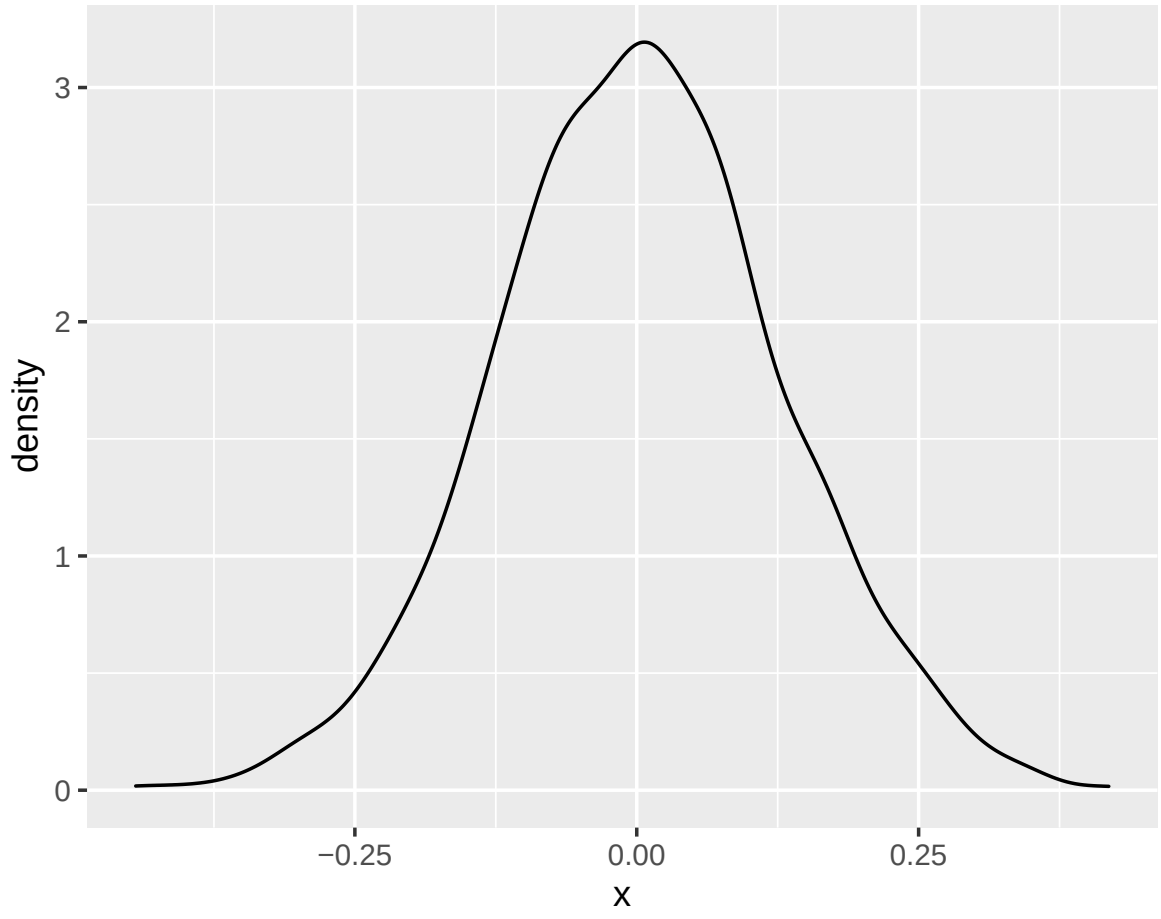
Ejemplo: ¿la desviación estándar de la mediana muestral?

- Suponga que observamos una muestra aleatoria $X_1, X_2, \ldots, X_n$ de una distribución de probabilidad desconocida f. Estamos interesados en decir algo sobre la mediana de la población, por lo que calculamos la mediana muestral $\tilde{X}$. $\rightarrow$ P: ¿Qué precisión tiene $\tilde{X}$ como estimador?
- Si conociéramos nuestra distribución F, podríamos tomar muestras de F y usar simulaciones para responder a nuestra pregunta.
- Sin embargo, sin conocimiento de la distribución, no podemos calcular la desviación estándar de nuestro estimador, es decir $\mathrm{SD}(\tilde{X})$.
- Ahí es donde entra en juego el método bootstrap.

Primero supongamos que conociéramos f, por ejemplo $X \sim N(0, 1)$. Luego, podemos tomar muestras repetidamente y calcular la desviación estándar de todas las medianas para obtener una estimación:

```
set.seed(123)
n = 101
B = 1000
estimator = rep(NA, B)
for (b in 1:B) {
    xs = rnorm(n)
    estimator[b] = median(xs)
}
sd(estimator)
```

```
## [1] 0.1259035
```


density
3
2
1
0
−0.25
0.00
0.25
x

De la simulación al bootstrapping (f desconocido)

- El método bootstrap utiliza los datos observados para estimar la *distribución empírica* $\hat{f}$, es decir, cada valor observado de x tiene una probabilidad de $1/n$.
- Una *muestra bootstrap* $X_1^*, X_2^*, \ldots, X_n^*$ es una muestra aleatoria extraída de $\hat{f}$.
- Una forma simple de obtener una muestra bootstrap es *extraer con reemplazo de* $X_1, X_2, \ldots, X_n$.
- **Nota**: Nuestra muestra bootstrap consiste de n miembros de $X_1, X_2, \ldots, X_n$ - algunos aparecen más de una vez, otros no aparecen en absoluto.

Compare la mediana muestral

```
set.seed(123)
n = 101
original = rnorm(n)
median(original)
```

```
## [1] 0.05300423
```

con la mediana de **una muestra bootstrap**:

```
boot1 = sample(x = original, size = n, replace = TRUE)
median(boot1)
```

```
## [1] -0.02854676
```

Sin embargo, extraer solo *una* de esas muestras no ayuda mucho.

El algoritmo bootstrap para estimar errores estándar

1. Extraer B muestras bootstrap: extraer con reemplazo de los datos originales.

2. Evaluar la estadística de interés en *cada una de las B muestras bootstrap* para obtener $\tilde{X}_b^*$ para la b-ésima muestra bootstrap.

3. Estime el error estándar al cuadrado:

$$\frac{1}{B-1}\sum_{b=1}^{B}(\tilde{X}_b^* - \frac{1}{B}\sum_{b=1}^{B}\tilde{X}_b^*)^2 \ ,$$

que es la desviación estándar empírica de las B estimaciones $\tilde{X}_b^*$, $b = 1, \ldots, B$.

Ilustración para el ejemplo de la mediana (con un bucle-for en R)

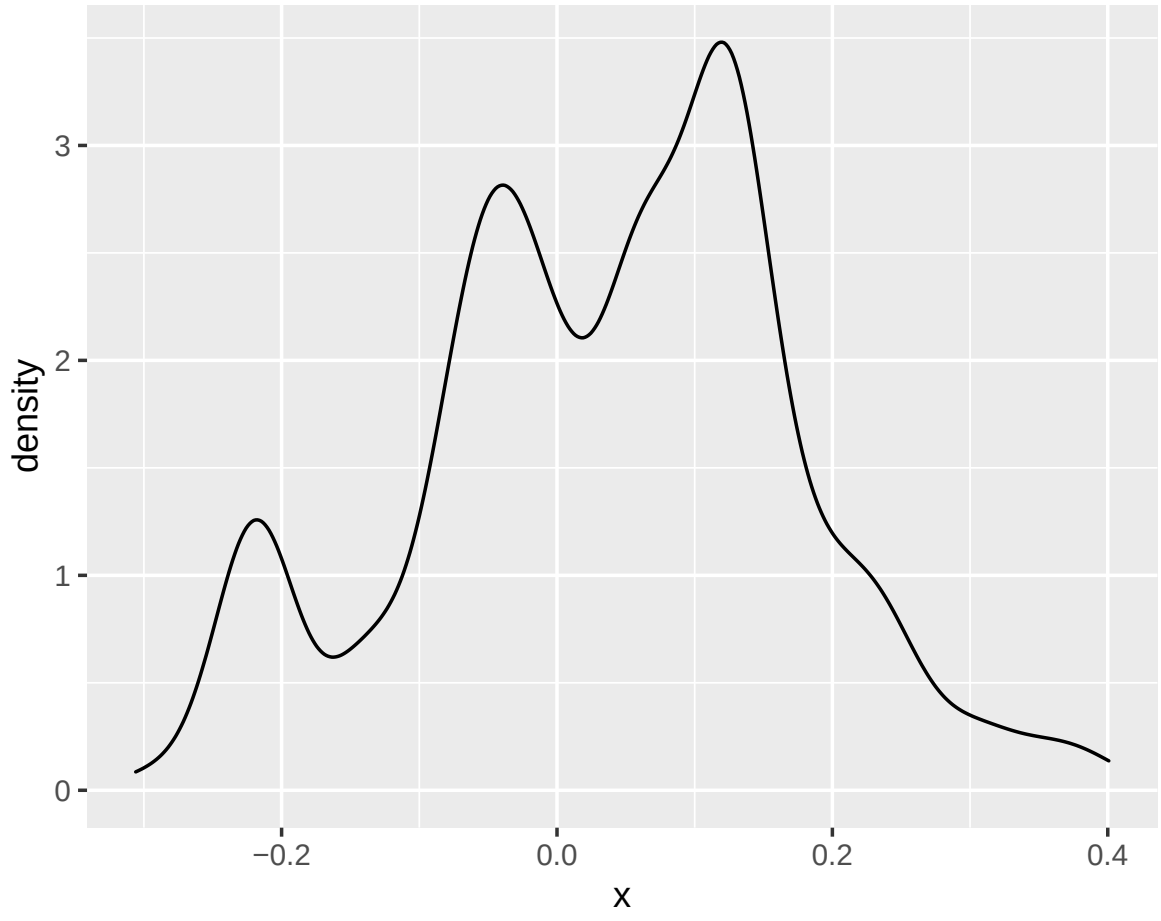
```
set.seed(123)
n = 101
original = rnorm(n)
median(original)
```

```
## [1] 0.05300423
```

```
B = 1000
estimator = rep(NA, B)
for (b in 1:B) {
    thisboot = sample(x = original, size = n, replace = TRUE)
    estimator[b] = median(thisboot)
}
sd(estimator)
```

```
## [1] 0.1365448
```

La distribución de las 1000 estimaciones muestreadas:



Alternativa: La función `boot` de la librería `boot`

```
library(boot)
boot.median = function(data, index) return(median(data[index]))
B = 1000
boot(original, boot.median, R = B)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = original, statistic = boot.median, R = B)
##
##
## Bootstrap Statistics :
##       original      bias    std. error
## t1* 0.05300423 -0.01602688   0.1310411
```

¿Con o sin reemplazo?

En bootstrapping tomamos muestras *con reemplazo* de nuestras observaciones.

P: ¿Qué pasa si en su lugar tomamos muestras *sin reemplazo*?

R: Entonces siempre obtendríamos la misma muestra, dado que el orden de los puntos muestrales no es importante para nuestro estimador.

(Nota al margen: en las pruebas de permutación, tomamos muestras sin reemplazo para obtener muestras bajo la hipótesis nula, un campo de investigación separado).

Ejemplo: regresión lineal múltiple

Asumimos, para la observación i:

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \varepsilon_i,$$

donde $i = 1, 2, ..., n$. El modelo se puede escribir en forma de matriz:

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon.$$

El estimador de mínimos cuadrados: $\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$ con $\text{Cov}(\beta) = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$.

Nota: Nuestras muestras de bootstrap también se pueden utilizar para hacer intervalos de confianza para los coeficientes de regresión o intervalos de predicción para nuevas observaciones. Esto significa que no tenemos que depender de suponer que los términos de error se distribuyen normalmente.

Un método relacionado: Bagging

Bagging (*bootstrap aggregation*) es un caso especial de *métodos agregados*.

- En el próximo Módulo hablaremos sobre bagging, que se basa en el bootstrapping y el hecho de que es posible reducir la varianza de una predicción tomando el promedio de muchos ajustes de modelo.
- Particularmente útil para métodos de estimación con varianza grande (como árboles de regresión).
- Idea:
  - Extraer B muestras bootstrap de los datos y entrenar el método con cada una de las b muestras con la finalidad de obtener $\hat{f}^{\star b}(x)$.
  - Para obtener una predicción, promediar todas las predicciones para obtener

$$\hat{f}_{bag}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^{\star b}(x) \ .$$

- Así, obtenemos un nuevo modelo que tiene una varianza menor que cada uno de los modelos individuales. Si las muestras de bootstrap fueran independientes (que por supuesto no lo son), la varianza (por lo tanto, el error de predicción) se reduciría en

$$\mathrm{Var}(\bar{X}) = \frac{\sigma^2}{B} \ .$$

- En realidad, la reducción de la varianza es menor. Para una correlación por pares ρ tendríamos $\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$.

- Los modelos que tienen poca capacidad de predicción (como podemos ver que puede suceder con árboles de regresión y clasificación) podrían beneficiarse enormemente del bagging.

Resumiendo

Mensajes para llevar a casa

- Utilice $k = 5$ o 10 veces la validación cruzada para la selección o evaluación del modelo.
- Utilice bootstrapping para estimar la desviación estándar de un estimador y comprender cómo se realiza.

Referencias adicionales

- Videos en YouTube por los autores de ISL, Capítulo 5, y los slides