Selección de variables

2da parte

Repaso

- Métodos de filtro
  - Uso de un criterio diferente al modelo final
  - Generalmente univariado
  - Rápidos (en general O(p))
- Wrappers
  - Buscar la mejor combinación con el modelo
  - Multivariado
  - Heurísticas de búsqueda. Back y For.

Métodos embebidos

- Los wrappers backward son potencialmente los mejores métodos de selección.
- Son computacionalmente muy pesados.
  - A cada paso construyen todos los clasificadores intermedios posibles para evaluarlos.
  - Para rankear p variables crean $O(p^2)$ modelos.
- La solución ideal sería un método backward, basado directamente en el modelo final, pero eficiente.

Métodos embebidos

- Para evaluar cuál es la próxima variable a eliminar, el wrapper back construye todos los modelos con una variable menos.
- Los evalúa a todos y “da un paso” en la dirección de máximo descenso del error.
- Se puede hacer algo parecido sin calcular todos los modelos target?

Métodos embebidos

- Lo que necesitamos conocer es la derivada del error respecto de cada variable.
  - O alguna aproximación a la derivada
  - Se las llama “medidas internas de importancia”
- Si la función error es razonablemente suave, dar el paso en la dirección de máximo descenso de la derivada debería ser lo mismo que el máximo descenso del error.

Métodos embebidos

- Recursive Feature Elimination (RFE):
  - Ajustar un modelo a los datos
  - Rankear las variables usando una medida interna de importancia.
    - Más importante es la que más empeora al modelo al ser eliminada
  - Eliminar la variable (o un grupo) con el ranking más bajo
  - iterar

El algoritmo RFE

Inputs:

Training set T

Set of p features $F = \{f_1, \ldots, f_p\}$

Ranking method $M(T, F)$

Outputs:

Final ranking R

Code:

Repeat for i in $\{1 : p\}$

Rank set F using $M(T, F)$

$f^* \leftarrow$ last ranked feature in F

$R(p - i + 1) \leftarrow f^*$

$F \leftarrow F - f^*$

Fig. 1. Pseudo-code for the Recursive Feature Elimination (RFE) algorithm.

RFE con SVM

Machine Learning, 46, 389–422, 2002

Gene Selection for Cancer Classification using Support Vector Machines

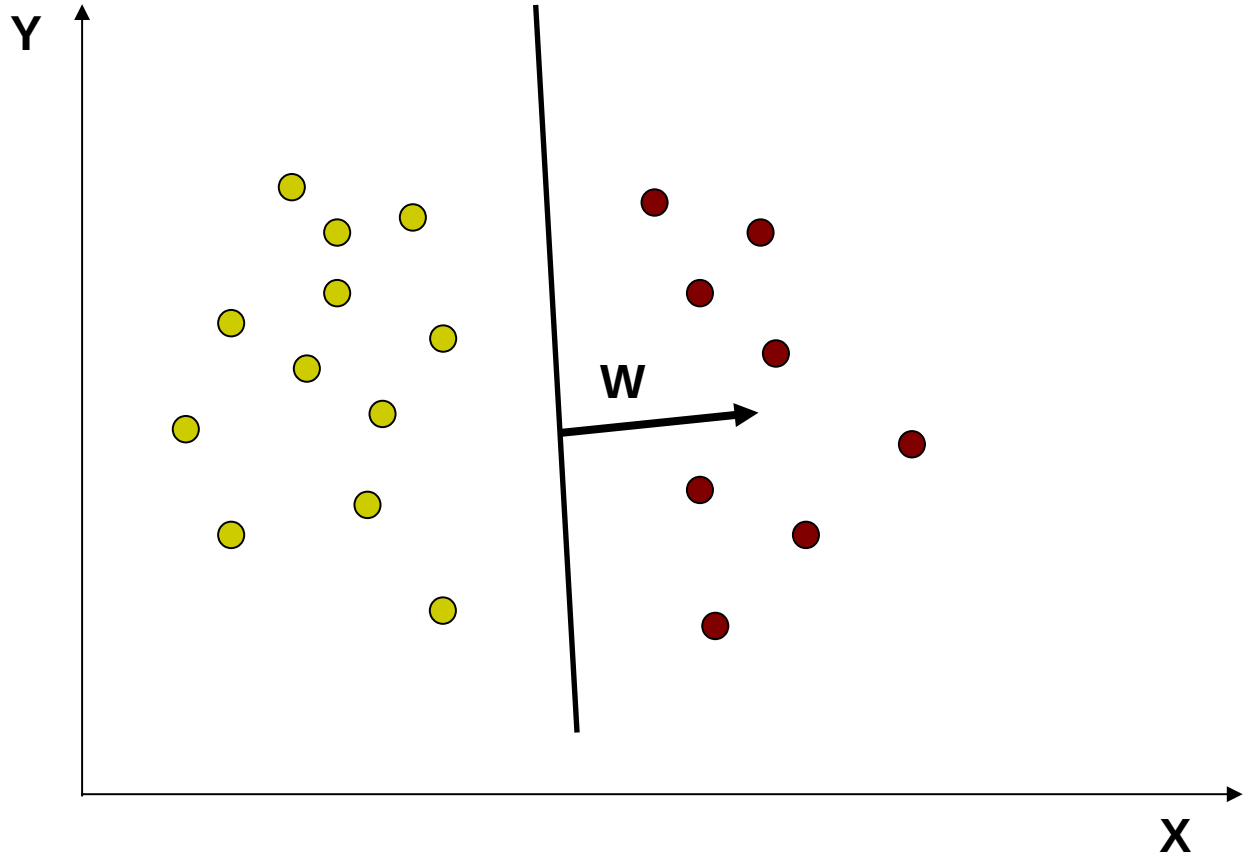
ISABELLE GUYON isabelle@barnhilltechnologies.com
JASON WESTON
STEPHEN BARNHILL
Barnhill Bioinformatics, Savannah, Georgia, USA

VLADIMIR VAPNIK vlad@research.att.com
AT&T Labs, Red Bank, New Jersey, USA

11000 citas 9/23

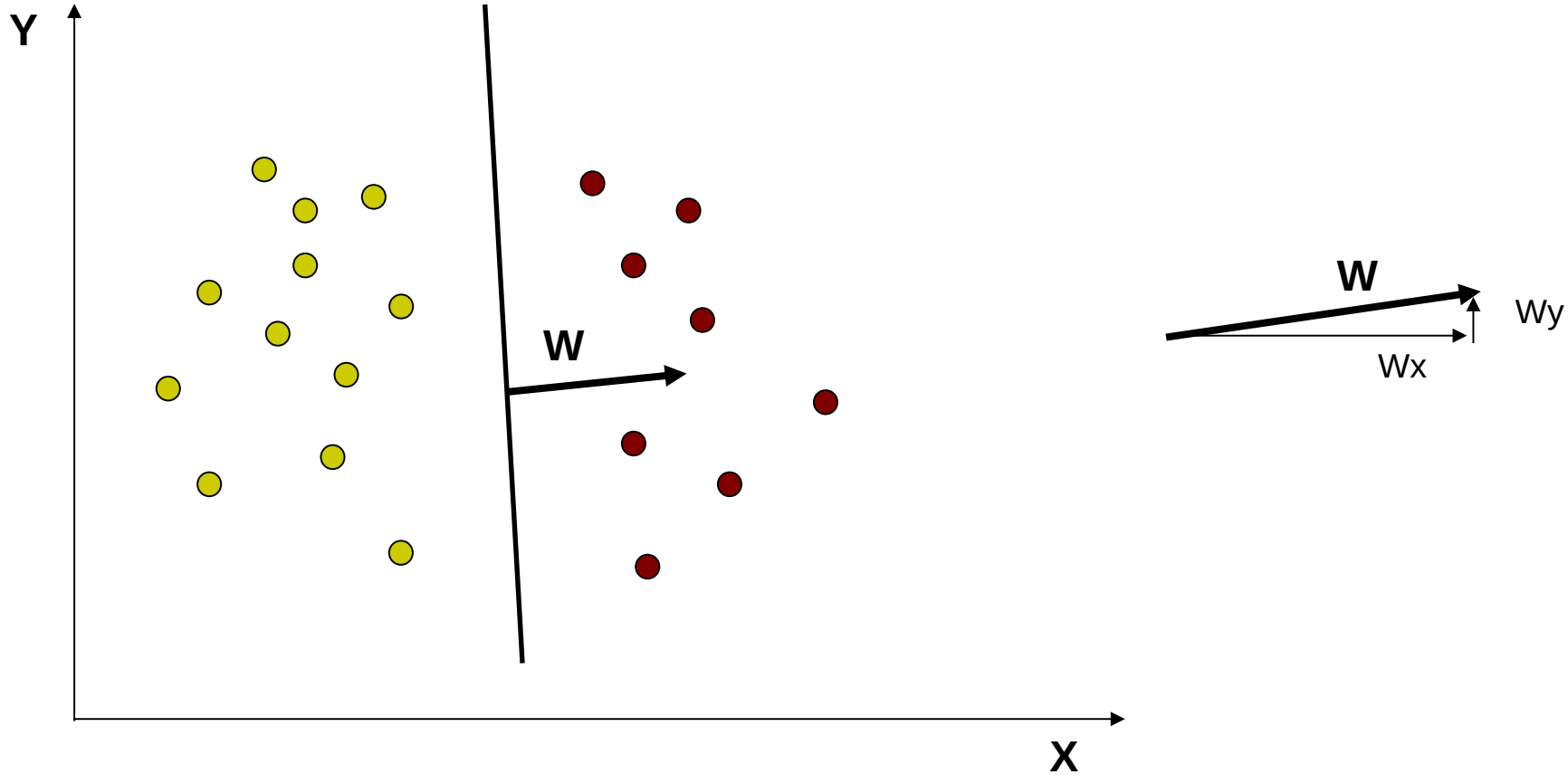
RFE con SVM

Medida de importancia SVM



RFE con SVM

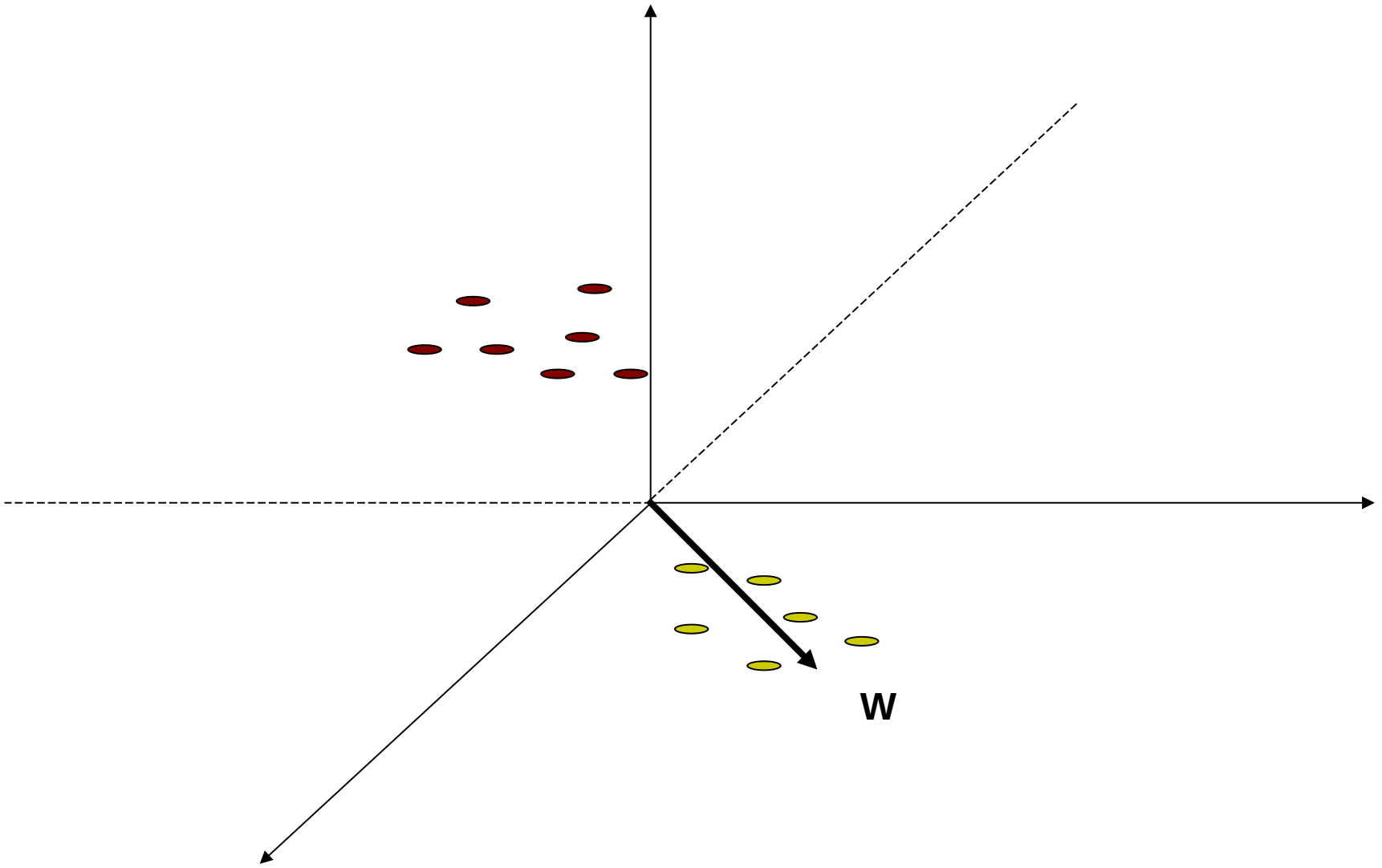
Medida de importancia SVM: componentes de W



Medidas de importancia

- Ejemplos de medidas de importancia:
  - SVM: componentes de W
  - Random Forest:
    - Shuffling OOB
    - Variación del GINI index.
  - LDA o PDA o Regresión logística: weights
  - Partial Least Squares (PLS): Scores

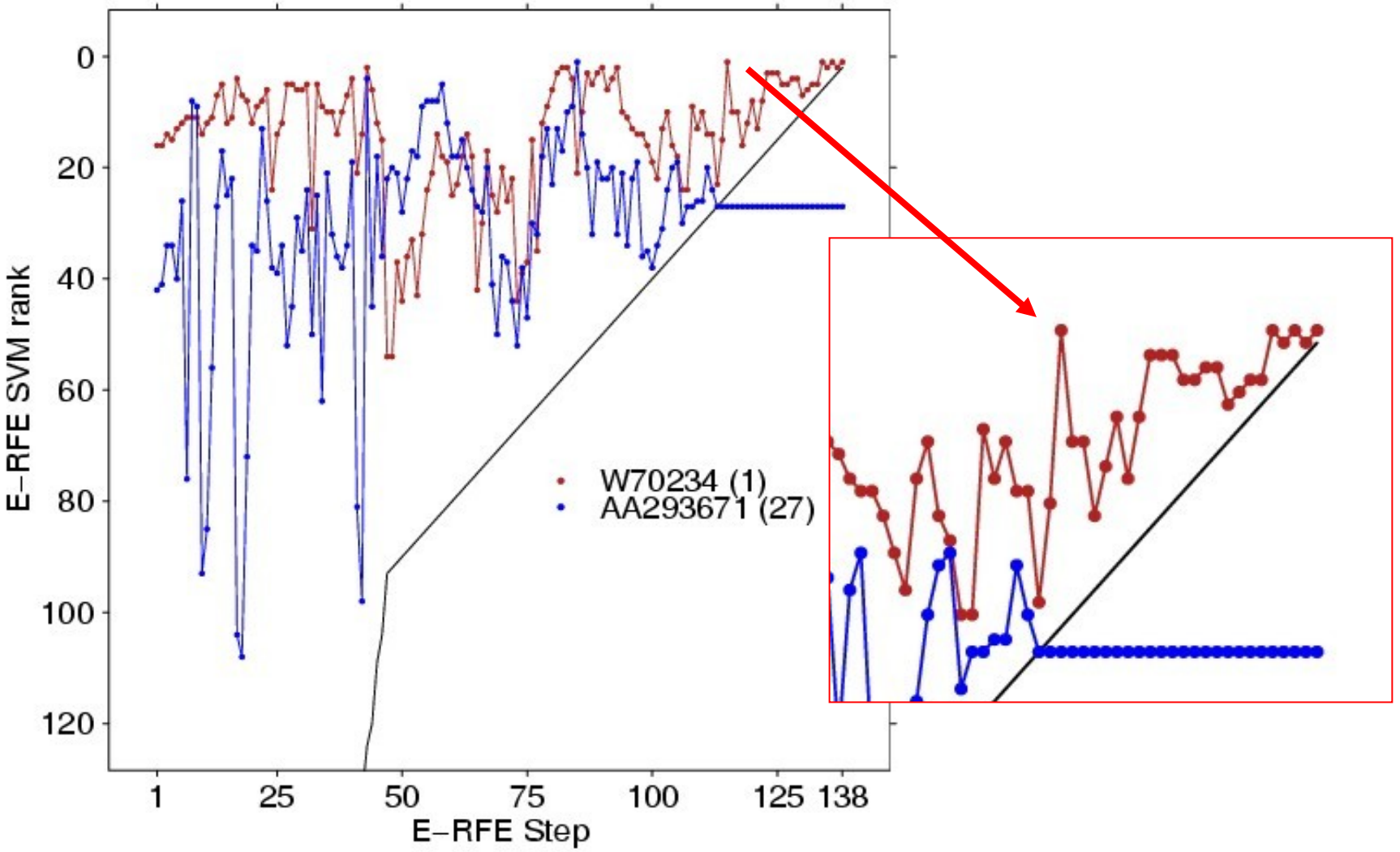
Problemas con el RFE



Problemas con el RFE

- Variables importantes pero correlacionadas
  - Si el modelo original usa las dos, las dos “comparten” la importancia.
  - En la práctica aparecen como menos importantes que otras variables.
  - Al eliminar una de ellas, la otra toma toda la importancia y suele subir bruscamente en el ranking (por esto es necesario iterar).

Problemas con el RFE



(d) Two highly correlated features.

Problemas con el RFE

- Variables importantes pero correlacionadas
  - Cuál es eliminada y cuál promocionada es casi chance.
  - Como resultado, el ranking de variables tiende a ser inestable.

Resumen: Métodos Embebidos o de Ranking

- Pros:
  - Rápidos
  - Efectivos
  - Entendibles
  - Más estables que los wrappers greedy
- Contras:
  - La importancia de las variables se estima, no se mide directamente
  - Problemas de inestabilidad con variables correlacionadas

Otras variantes

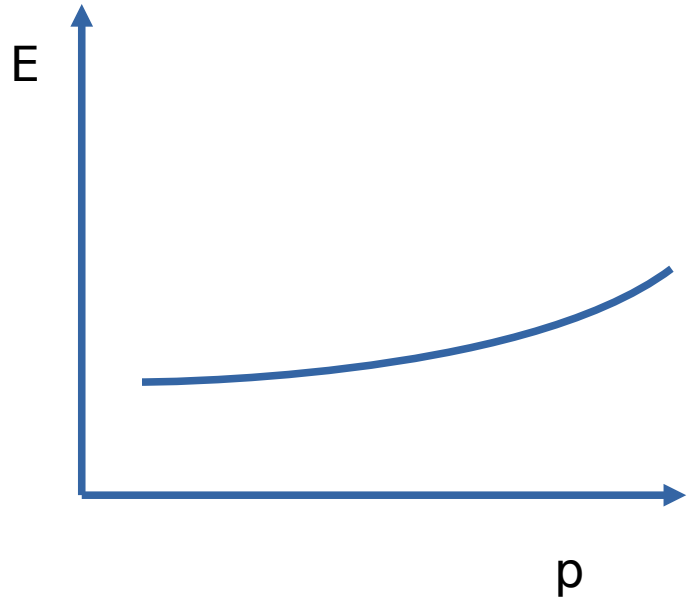
- Algunos métodos de aprendizaje incorporan la selección directamente.
  - Ejemplo: Arboles de decisión (c4.5) hacen una selección greedy forward
- Suelen no ser muy eficientes

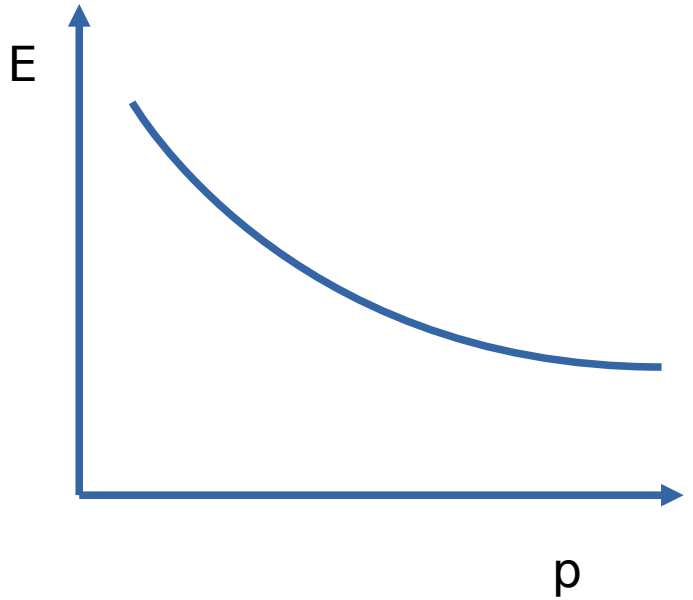
Otras variantes (no las vemos)

- En vez de seleccionar variables (problema combinatorio), por que no darle un peso estadístico a cada variable?
- Muchos métodos. La mayoría basado en el wrapper approach.
- Heurística: se optimizan los pesos junto con todos los otros parámetros del método de aprendizaje.

Evaluando las selecciones

- Quiero saber cuántas variables me tengo que quedar
- Es muy útil en general tener idea de cómo cambia el error de predicción al eliminar variables.





Evaluando las selecciones

- Es muy útil tener idea de cómo cambia el error de predicción al eliminar variables.
- Idea base:
  - Aplico el wrapper o filtro o RFE a mi problema.
  - Al ir eliminando variables, controlo el error. Busco el mínimo.
  - Qué error?

Evaluando las selecciones

- Idea base:
  - Aplico el wrapper o filtro o RFE a mi problema.
  - Al ir eliminando variables, controlo el error. Busco el mínimo.
  - Qué error? En principio se usó cross-validation
    - Divido los datos internamente.
    - Busco la variable a eliminar usando una parte de los datos, y calculo el error del modelo sin esa variable con otra parte de los datos. Repito y promedio.
    - Itero

Evaluación incorrecta

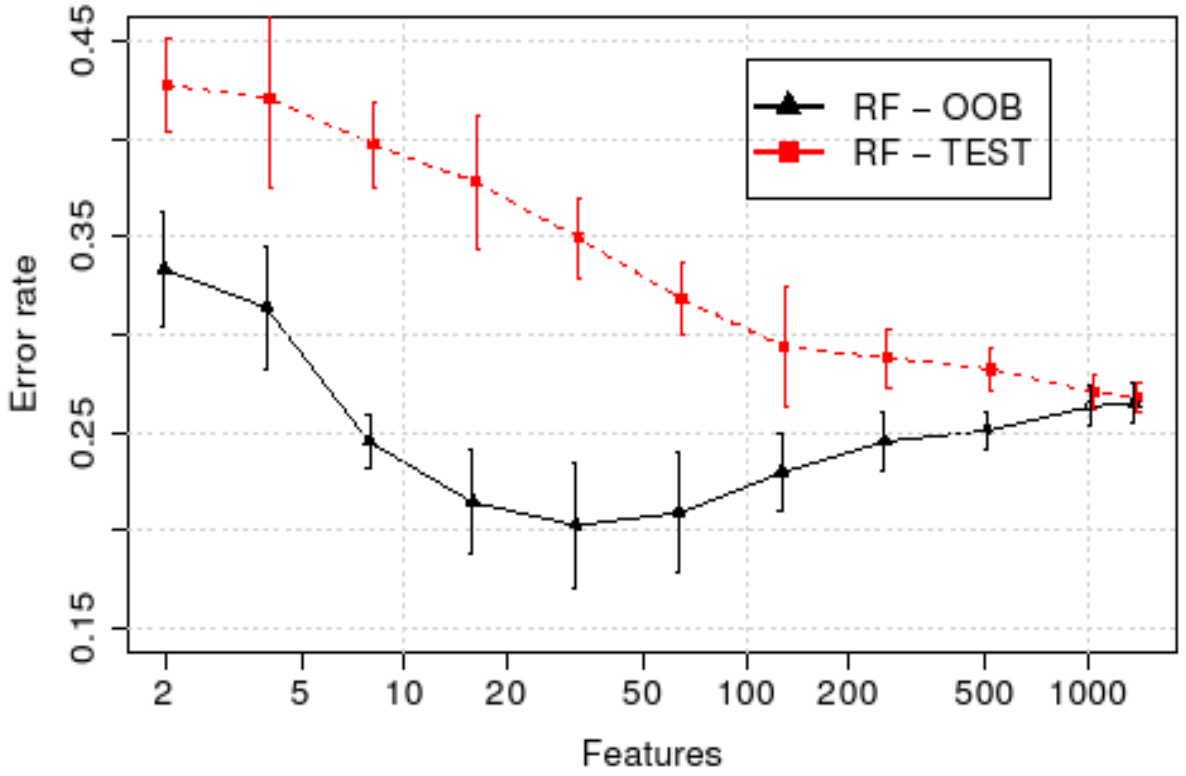
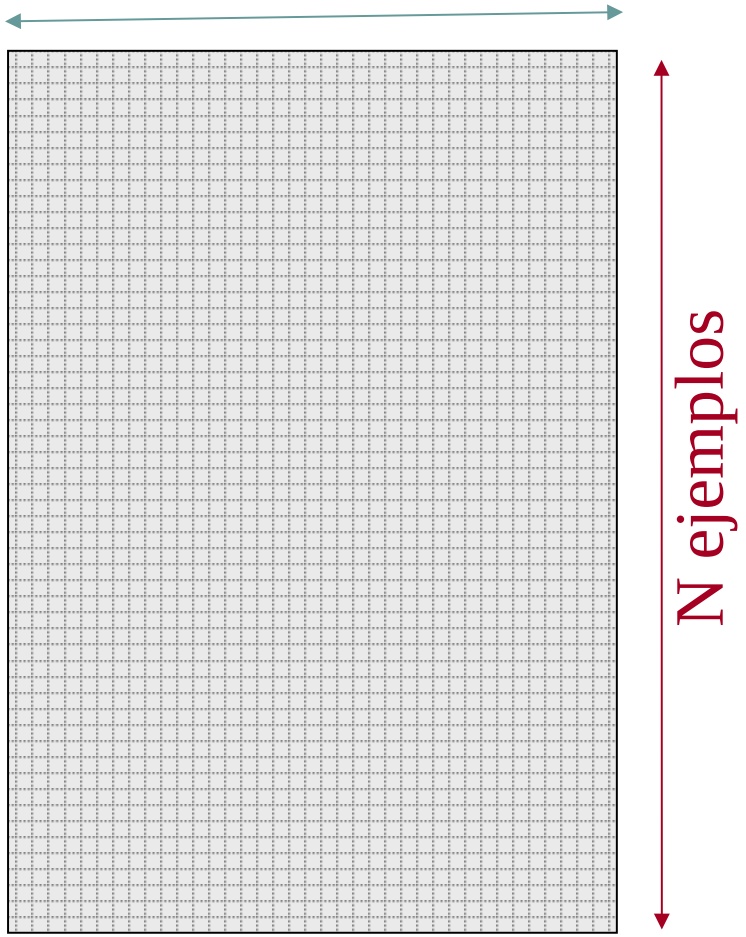


Figure 1: Error levels for OOB and real test set for RF-RFE on random data. We generated a full random dataset with 1380 random inputs with N (0, 1) distribution, with random labels, 131 of class 0 and 47 of class 1. Bayes error: 47/178=0,26. Error lines show one standard deviation.

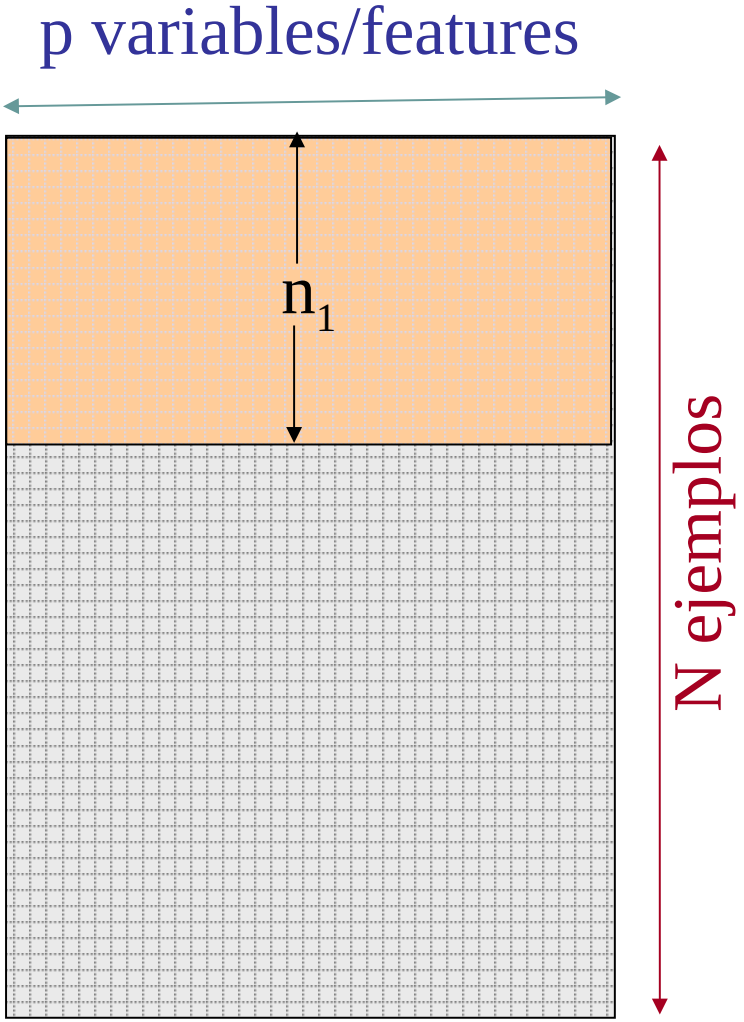
Evaluando las selecciones



Partir los datos en 3 partes:

training, validación y test set.

Evaluando las selecciones

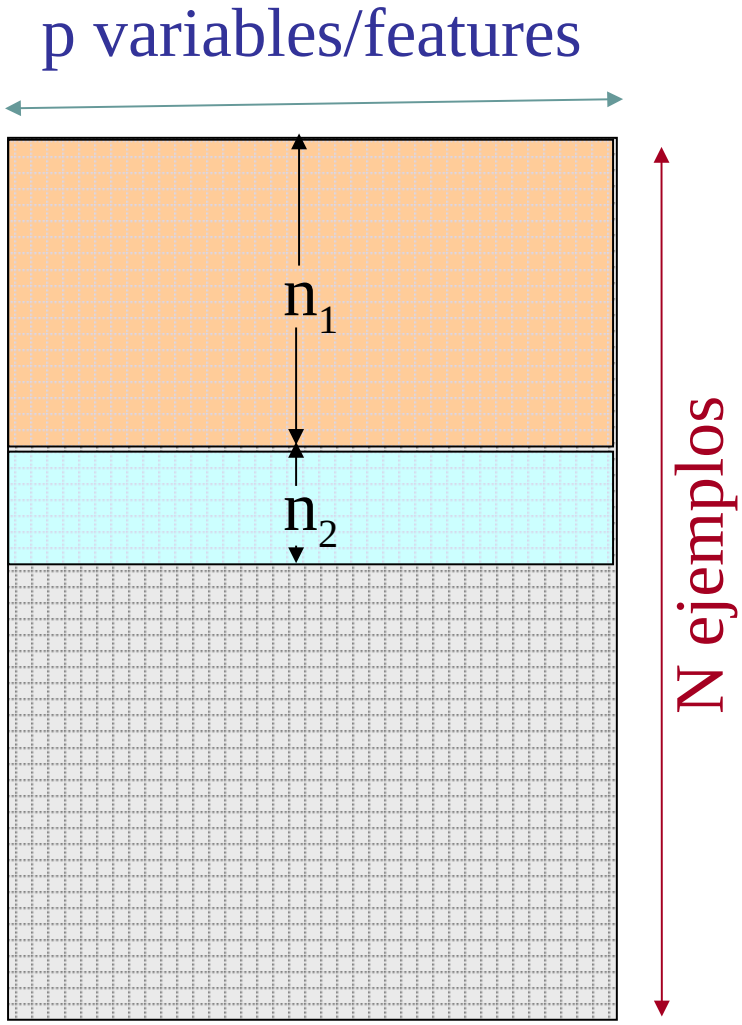


Partir los datos en 3 partes:

training, validación y test set.

1) Para cada subconjunto de variables entrenar un modelo en el training set.

Evaluando las selecciones



Partir los datos en 3 partes:

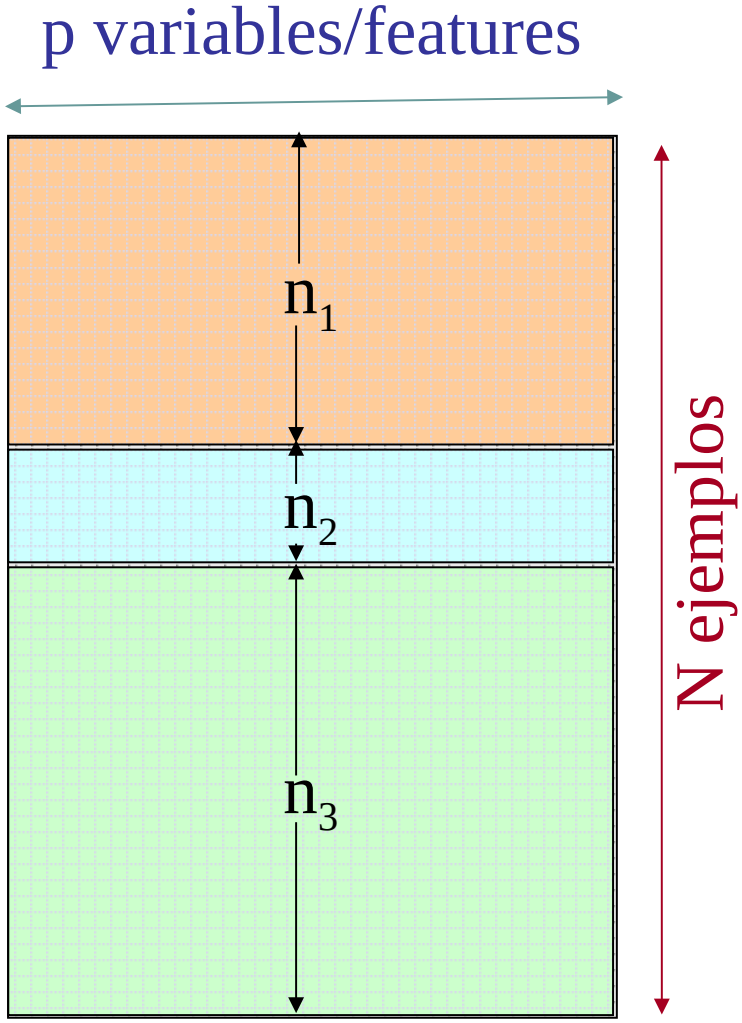
training, validación y test set.

1) Para cada subconjunto de variables entrenar un modelo en el training set.

2) Elegir el subconjunto de variables que muestra la mejor performance en el validation set.

- Se puede usar cross-validation para tener una mejor estima.

Evaluando las selecciones



Partir los datos en 3 partes:

training, validación y test set.

1) Para cada subconjunto de variables entrenar un modelo en el training set.

2) Elegir el subconjunto de variables que muestra la mejor performance en el validation set.

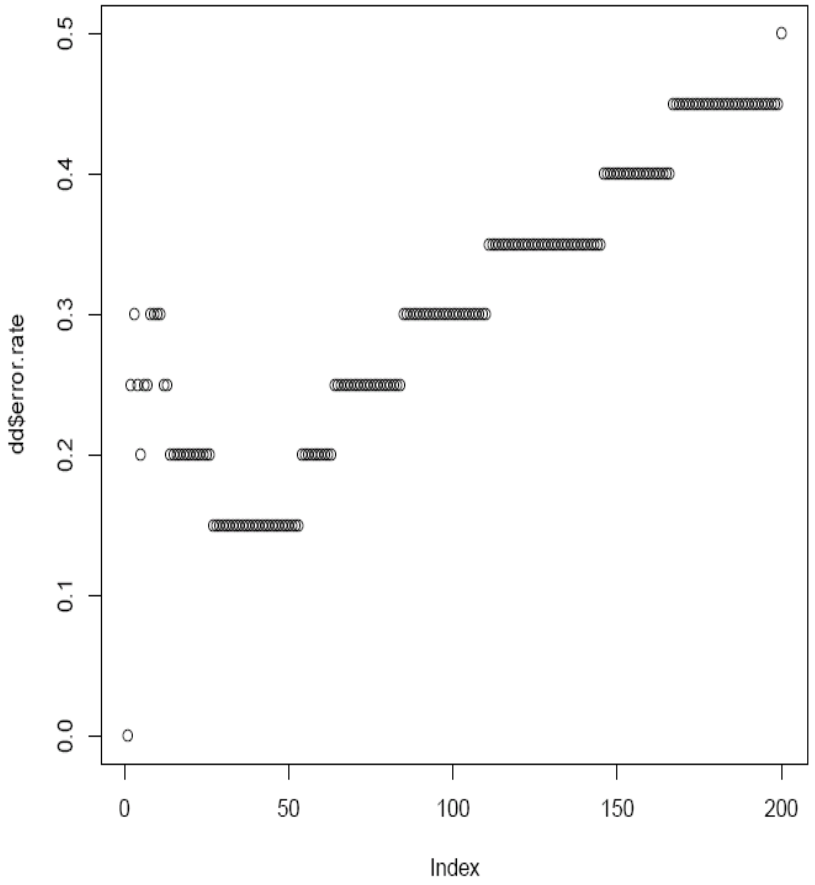
- Se puede usar cross-validation para tener una mejor estima.

3) Medir el error verdadero en el test set.

Si buscamos mucho…

- Ejemplo: prueba_rnd.R
- Genero ruido, n=20, p=200 (Bayes=0.5)
- Los datos son ruido uniforme en todas las dimensiones
- Clasificador fijo, hiperplano apuntando al (1,1,...1)
- Busqueda greedy con cross-validation

Si buscamos mucho…

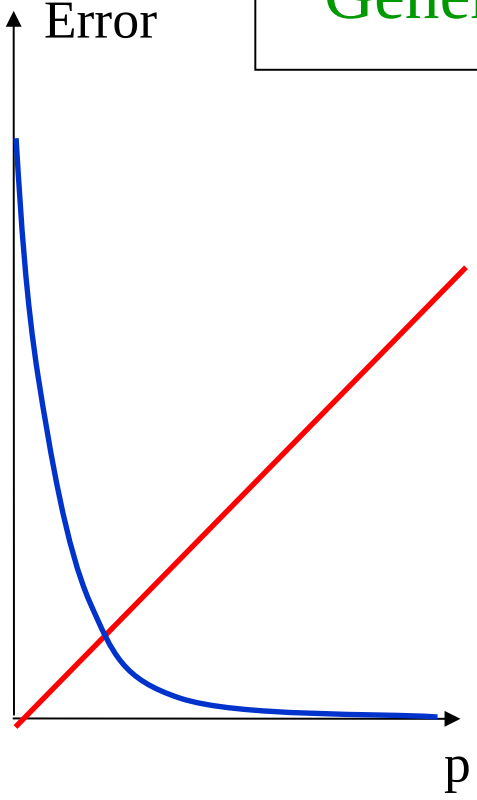


- Error con 20 variables=0.15???
- Estoy haciendo tests múltiples!

Complejidad en la selección

Con alta probabilidad (Vapnik):

$$\text{Generalization\_error} \leq \text{Validation\_error} + \varepsilon(C/n_2)$$



Método	Número de evaluaciones	Complejidad C
Búsqueda exhaustiva	2^P	P
Selecciones anidadas "ranking"	P(P+1)/2 o P	log P

n_2: número de ejemplos de *validation*,

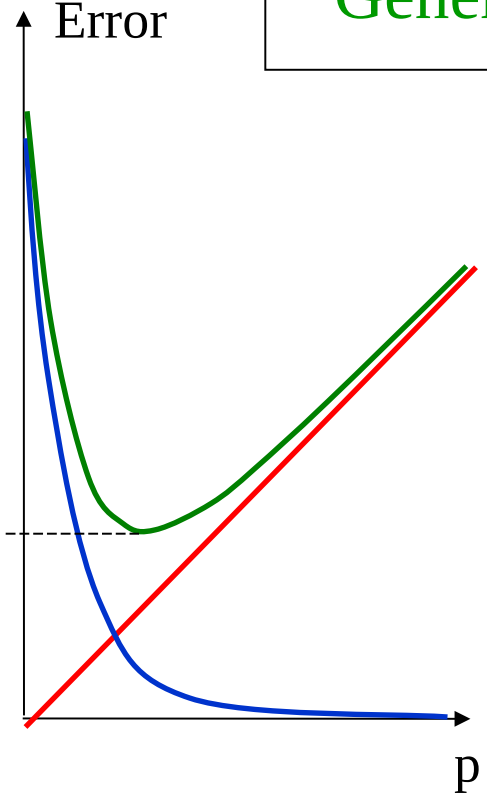
P: número total de variables,

p: tamaño del subconjunto.

Complejidad en la selección

Con alta probabilidad (Vapnik):

$$\text{Generalization\_error} \leq \text{Validation\_error} + \varepsilon(C/n_2)$$



Método	Número de evaluaciones	Complejidad C
Búsqueda exhaustiva	2^P	P
Selecciones anidadas "ranking"	P(P+1)/2 o P	log P

n_2: número de ejemplos de *validation*,

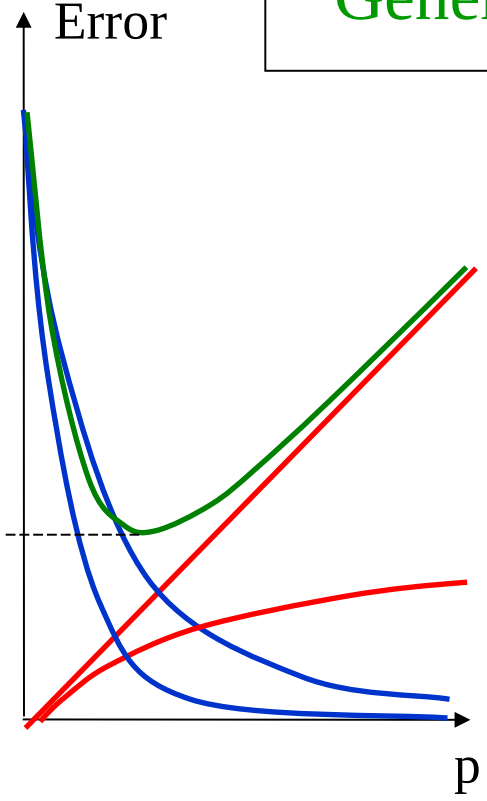
P: número total de variables,

p: tamaño del subconjunto.

Complejidad en la selección

Con alta probabilidad (Vapnik):

$$\text{Generalization\_error} \leq \text{Validation\_error} + \varepsilon(C/n_2)$$



Método	Número de evaluaciones	Complejidad C
Búsqueda exhaustiva	2^P	P
Selecciones anidadas "ranking"	P(P+1)/2 o P	log P

n_2: número de ejemplos de *validation*,

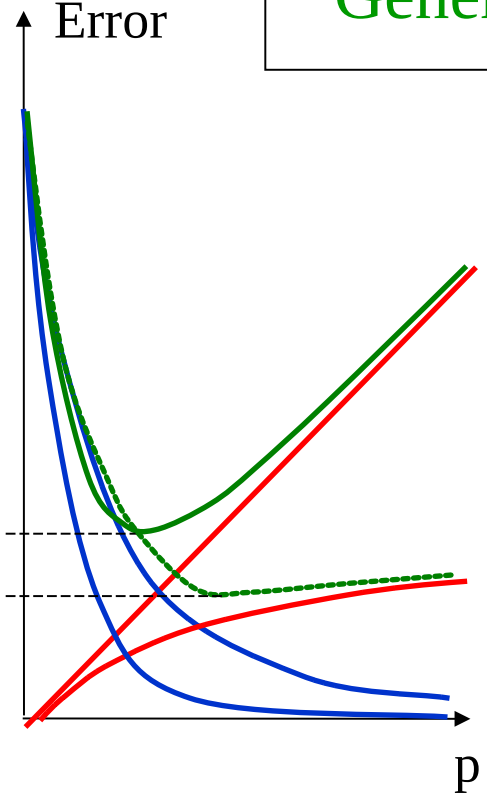
P: número total de variables,

p: tamaño del subconjunto.

Complejidad en la selección

Con alta probabilidad (Vapnik):

$$\text{Generalization\_error} \leq \text{Validation\_error} + \varepsilon(C/n_2)$$



Método	Número de evaluaciones	Complejidad C
Búsqueda exhaustiva	2^P	P
Selecciones anidadas "ranking"	P(P+1)/2 o P	log P

n_2: número de ejemplos de *validation*,

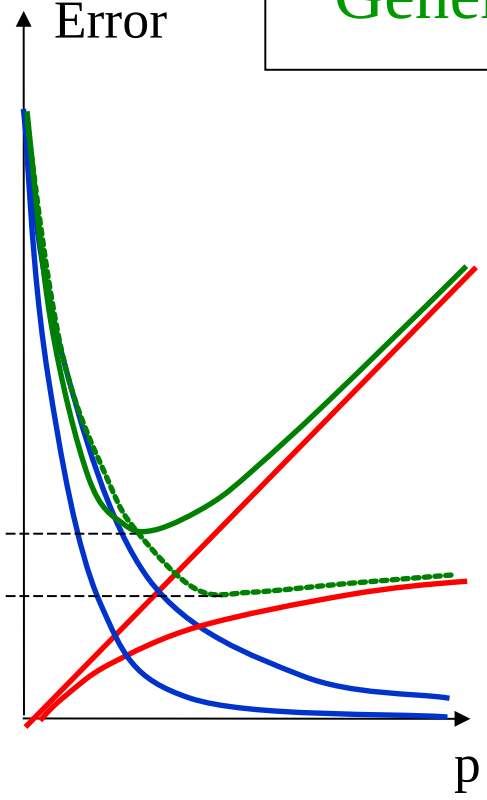
P: número total de variables,

p: tamaño del subconjunto.

Complejidad en la selección

Con alta probabilidad (Vapnik):

$$\text{Generalization\_error} \leq \text{Validation\_error} + \varepsilon(C/n_2)$$



Método	Número de evaluaciones	Complejidad C
Búsqueda exhaustiva	2^P	P
Selecciones anidadas "ranking"	P(P+1)/2 o P	log P

n_2: número de ejemplos de *validation*,

P: número total de variables,

p: tamaño del subconjunto.

Acotar C mejora el resultado!

Método sugerido

- Usar el train para elegir las variables.
- Usar validación independiente para determinar la cantidad de variables a retener.
- Hacer una selección final con train+validación.
- Estimar el error con el test set (pero no elegir nada).

Problema abierto

- Tenemos conjuntos chicos y necesitamos tests independientes. Usamos repeticiones para reducir la varianza de los resultados.
- Cada repetición da una lista distinta de variables. Cuales son las variables importantes de verdad?

Práctico 2

- En R, armar el RFE, Backward Greedy y filter.
  - Hay funciones de importancia ya disponibles para los 3.
- Aplicar a algunos ejemplos artificiales y analizar los resultados.
- Dataset real: limpiarlo, graficarlo, dar información sobre variables relevantes