Big Data y Minería de Datos

Unidad 3. Aprendizaje computacional

U3.2. Aprendizaje supervisado

Autores: José Antonio Ruipérez Valiente (jruiperez@um.es),
Mariano Albaladejo González (mariano.albaladejog@um.es) y
Manuel Jesús Gómez Moratilla (manueljesus.gomezm@um.es)

UNIVERSIDAD DE
MURCIA

Unidad 3. Aprendizaje computacional
Estructura

UNIVERSIDAD DE MURCIA

- Fundamentos del aprendizaje supervisado
- Selección de un modelo
- Evaluación de modelos
- Naive Bayes
- K-Vecinos Próximos

Big Data y Minería de Datos
Contenidos

UNIVERSIDAD DE MURCIA

Introducción a la minería de datos

Técnicas de inteligencia artificial

Reglas
Fundamentos lógicos
Sistemas basados en reglas
Reglas de asociación

Aprendizaje computacional
Fundamentos
Regresión y clasificación
No supervisado

Procesado del lenguaje natural
Fundamentos
Representación y tópicos
Sentimientos y NER

Introducción al big data

UNIVERSIDAD DE MURCIA

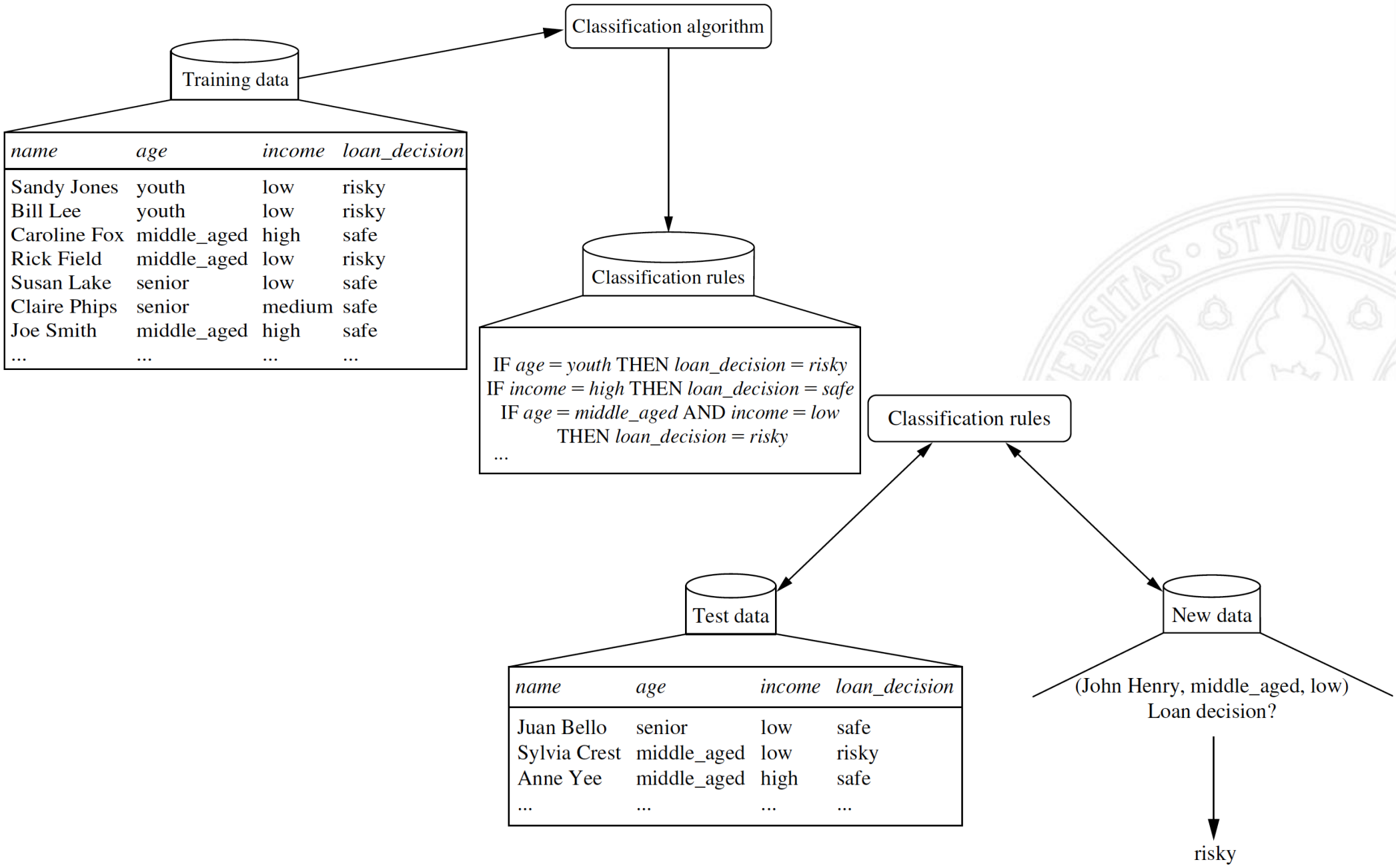
Unidad 3. Aprendizaje computacional

FUNDAMENTOS DEL APRENDIZAJE SUPERVISADO

Unidad 3. Aprendizaje computacional
El problema básico vs el supervisado

- Encontrar una función *f(X)* que es desconocida en base a una entrada
- En el problema básico teníamos un vector de datos de entrada
- En el problema supervisado, tenemos un vector de parejas de entrada tal que*:* $(x_1, y_1), (x_2, y_2), \ldots (x_N, y_N)$
- Se llama a la salida y_i como verdad fundamental (ground truth)
- Cada pareja fue generada por una función desconocida $y = f(x)$
- Función hipótesis (*h*): hipótesis *h* que aproxime a *f*
- Espacio de hipótesis ($\mathcal{H}$): *h* es un modelo de los datos, obtenido de todos los posibles modelos, $h \in \mathcal{H}$
- La función *h* puede tomar formas muy variadas, como un polinomio de grado 3, un código en Python o un modelo de reglas con CLIPS

Unidad 3. Aprendizaje computacional
El proceso de clasificación

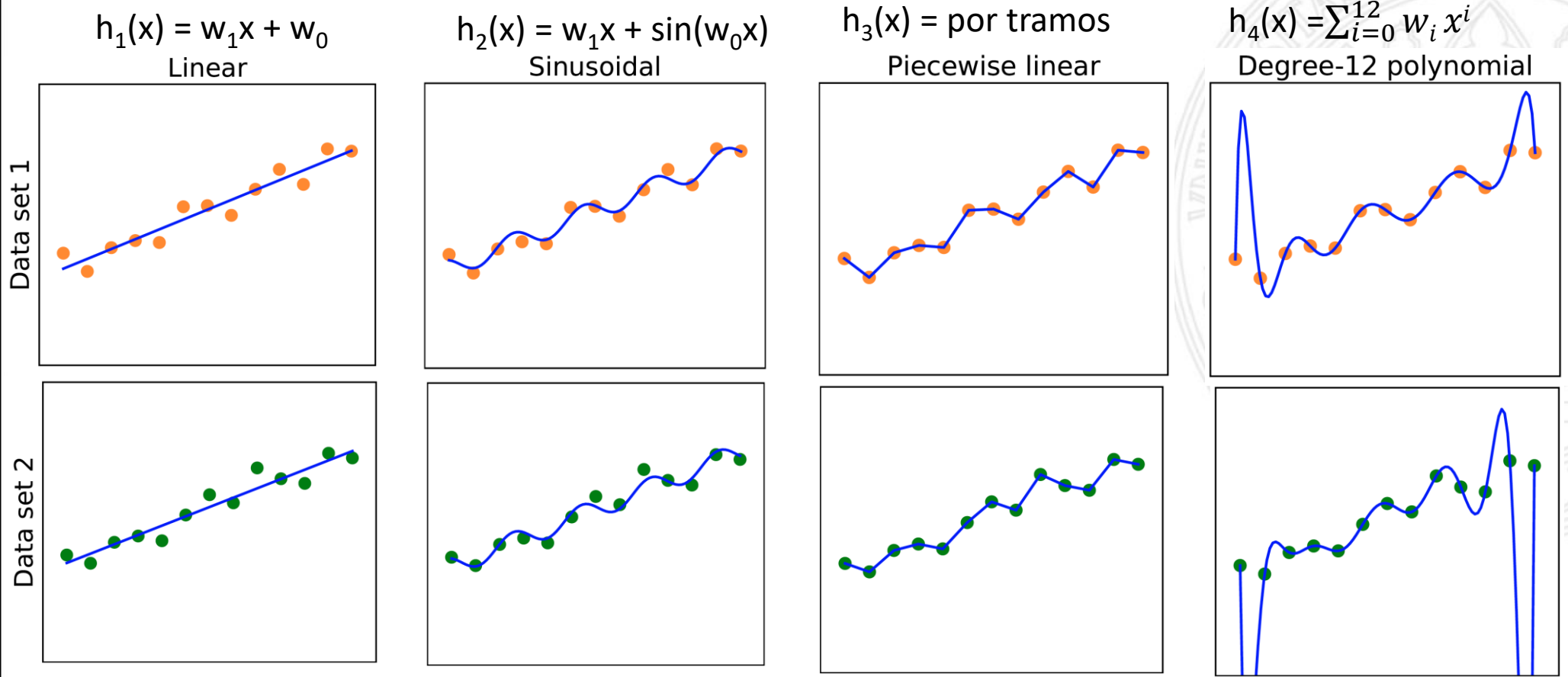


UNIVERSIDAD DE MURCIA

- ¿Cómo elegir el modelo dentro del espacio de hipótesis?
  - Conocimiento previo, visualizar/analizar datos, entrenamiento…
- Idealmente, el modelo sería consistente en el que:
  - $h\ consistente \leftrightarrow \forall\ (x_i, yi) \in h(x_i) = yi$
  - Poco realista, especialmente para variables continuas
  - Además, no habrá visto los datos del juego de test
- Buscaremos el *h* que mejor se ajuste a *f*, generándolo usando el dataset de entrenamiento
- Diremos que *h* es una **buena generalización** si predice de **forma satisfactoria** las **salidas** del dataset de **test**
- La verdadera medida de calidad se basa en la generalización sobre el dataset de test

UNIVERSIDAD DE MURCIA

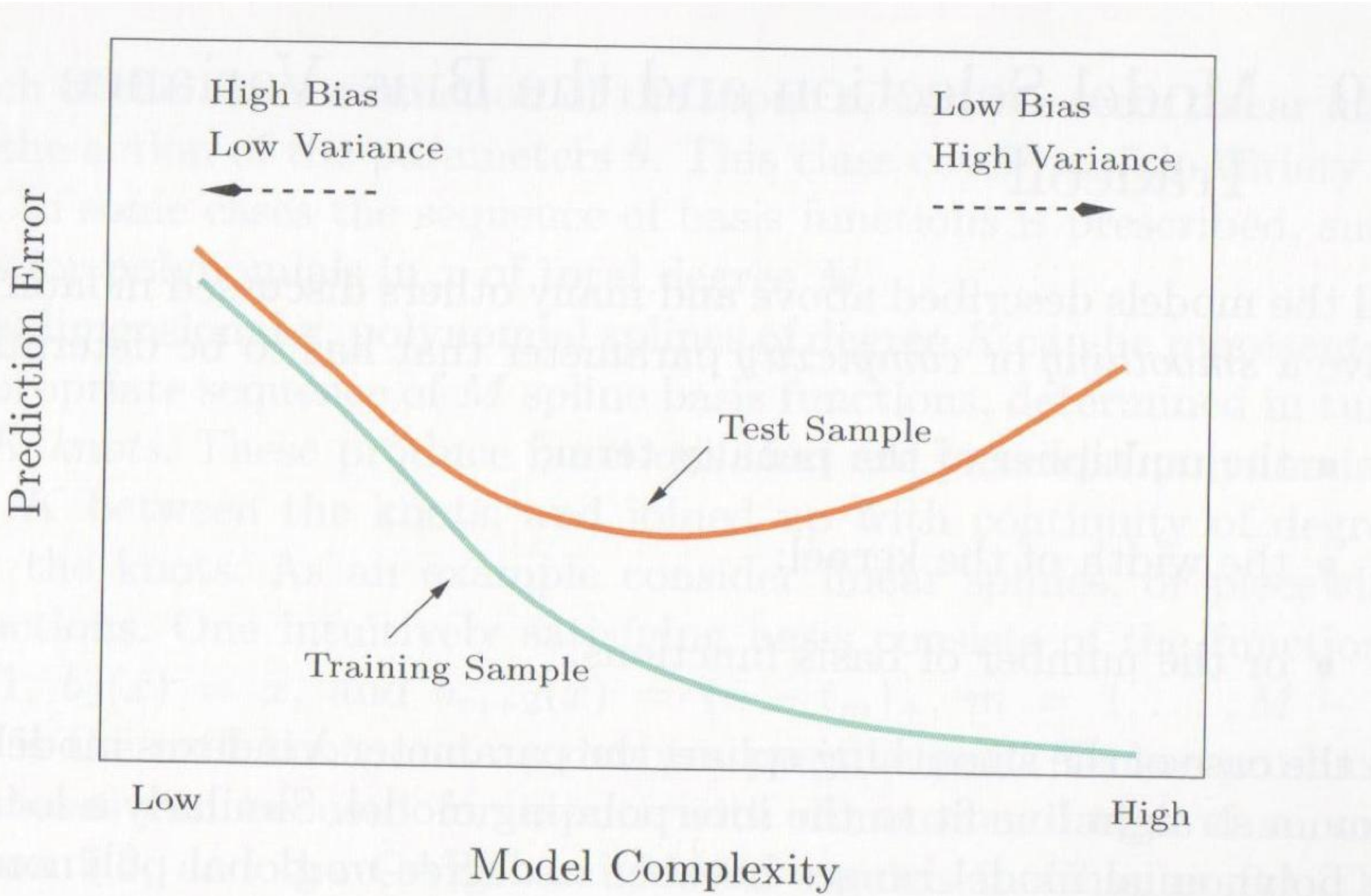
- Podemos utilizar varias funciones para ajustar el modelo *h*
- ¿Cuál es la mejor hipótesis?
- Podemos extraer una nueva muestra de datos X también proveniente de la función *f* y repetir el proceso



- Una manera de evaluar los modelos es en función del sesgo que imponen y la varianza que producen
- **Sesgo**: Tendencia de la predicción hipotética a desviarse del valor esperado en diferentes datasets de entrenamiento
  - Puede venir impuesta por la función escogida y su expresividad
  - Por ejemplo, las líneas pueden ajustarse poco y la función por tramos mucho
- **Subajuste**: cuando no logra ajustarse a los patrones de los datos de entrenamiento
- **Varianza**: cantidad de cambio en la hipótesis debida a la fluctuación de los datos de entrenamiento
  - Por ejemplo, ¿cuáles tendrán mayor varianza en el ejemplo?
- **Sobreajuste**: cuando la función se ajusta demasiado a los datos entrenamiento y no generaliza bien con nuevos datos

- La elección entre **algoritmos complejos** con **sesgos pequeños** que se **ajustan** a los datos de **entrenamiento bien** vs. algoritmos con **baja varianza** que **generalizan mejor**
- No es posible aprender sin ese sesgo, por lo que hay que escoger el sesgo inductivo más adecuado
- Se consiguen buenas generalizaciones si se ajusta bien la complejidad modelo a la complejidad subyacente de los datos
  - Si nos quedamos cortos: subajuste (alto error en train y test)
  - Si nos pasamos: sobreajuste (bajo error en train y alto en test)
- La virtud es una solución de compromiso entre:
  - La complejidad de la hipótesis
  - El tamaño del conjunto de entrenamiento
  - El error de generalización de los nuevos casos

Unidad 3. Aprendizaje computacional
Balance entre el sesgo-varianza



UNIVERSIDAD DE MURCIA

Unidad 3. Aprendizaje computacional

SELECCIÓN DE UN MODELO

- Formalmente esta búsqueda de la hipótesis se puede formular de la siguiente manera:

$$\mathrm{H}^{*} = \operatorname{argmax}_{\mathrm{hi} \in \mathcal{H}} \mathrm{P}(\mathrm{hi}|\mathrm{data})$$

donde $\mathrm{P}(\mathrm{hi}|\mathrm{data})$ es la probabilidad de encontrar una *h* que se ajuste a los datos

- Aplicando Bayes:

$$\mathrm{H}^{*} = \operatorname{argmax}_{\mathrm{hi} \in \mathcal{H}} \mathrm{P}(\mathrm{data}|\mathrm{h_i})\mathrm{P}(\mathrm{h})$$

$\mathrm{P}(\mathrm{data}|\mathrm{h_i})$ es la probabilidad de que los datos se ajusten a *h*

$\mathrm{P}(h)$ es la probabilidad de que seas una buena función

Por ejemplo, un polinomio 12 puede tener una probabilidad menor que una función lineal

- Establecer $\mathcal{H}$ como cualquier programa de ordenador
  - Balance expresividad de hipótesis y complejidad computacional
  - Por esta razón, se suele usar modelos simples

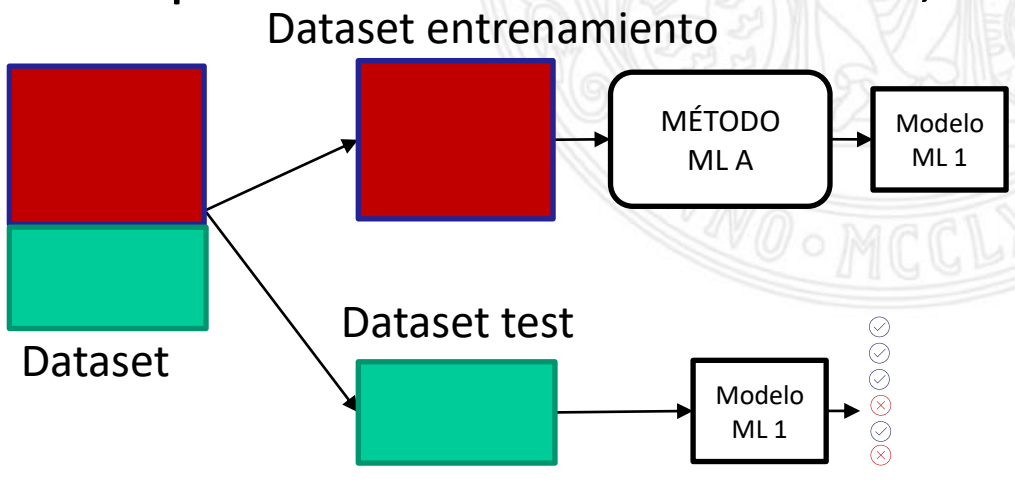
Unidad 3. Aprendizaje computacional
Proceso de entrenamiento para elegir el modelo

- Queremos un modelo que minimice el error, de forma simplificada, *h(x) = y* para cualquier *(x, y)*
- Por una parte, tendremos que elegir el tipo de método ($\mathcal{H}$)
  - Por ej, reglas, árboles, redes, k-vecinos más cercanos…
  - A esto lo vamos a llamar selección de la clase de modelo
- Por otra parte, una vez elegido el tipo de método, tendremos que ajustar sus parámetros (*h*)
  - Por ej, en árboles los pesos de las conexiones, el tamaño, etc…
  - A esto lo vamos a llamar selección de la mejor hipótesis en ese espacio de modelos
  - Esto lo haremos mediante el proceso de entrenamiento

Unidad 3. Aprendizaje computacional
Validación simple

UNIVERSIDAD DE MURCIA

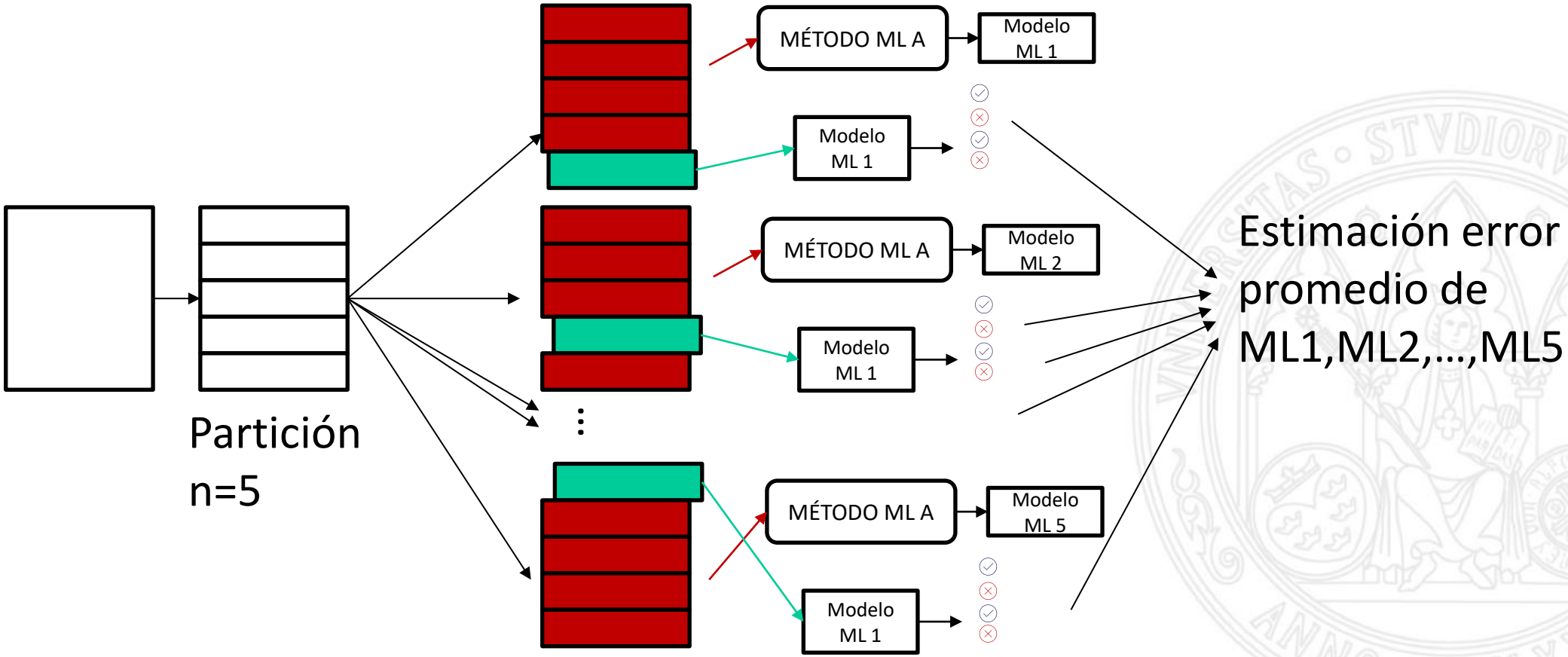
- Selección del modelo, su ajuste y la estimación del error
  - Óptimo para casos donde hay gran cantidad de datos
- El dataset de entrenamiento: Se usa para entrenar los modelos candidatos y ajustar sus parámetros
- El dataset de validación: Se usa para evaluar los modelos candidatos y elegir el mejor
- El dataset de test: Se usa para obtener una evaluación final no sesgada del mejor modelo
  - **NUNCA** debe usarse como parte del proceso de entrenamiento, sólo para la evaluación final



Unidad 3. Aprendizaje computacional
Validación cruzada (k-fold cross-validation)

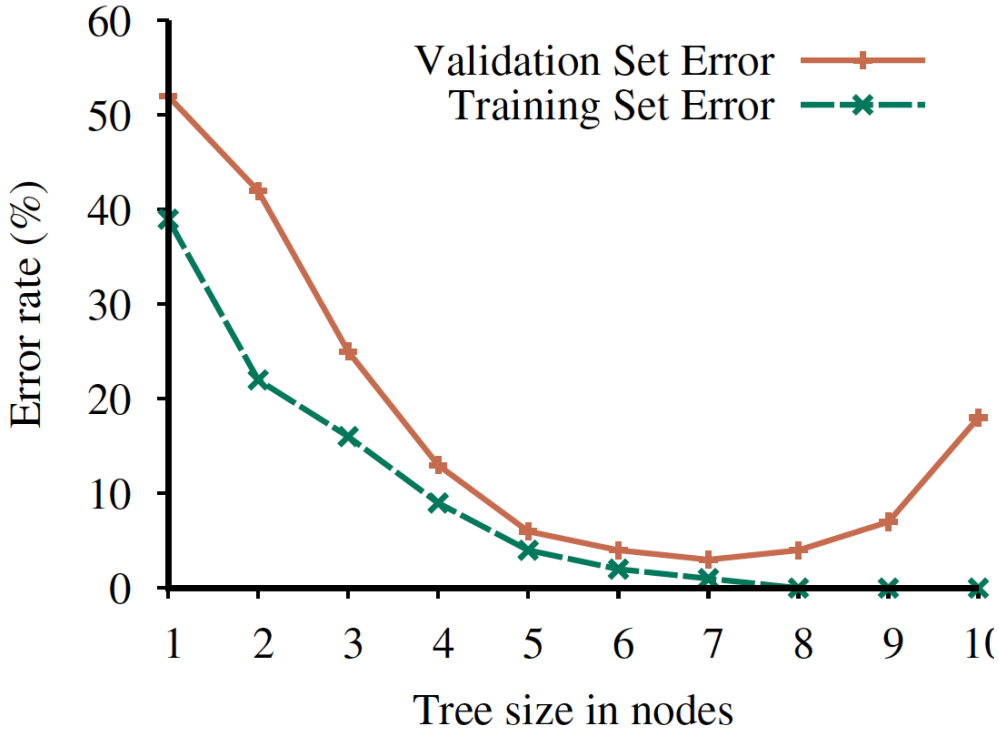
- Cada ejemplo hará tarea doble: entrenamiento y validación, pero no al mismo tiempo. Pasos a seguir:
  - División en *k* partes iguales y se hacen *k* rondas:
    1. Se retiene una parte para validar y el resto de los ejemplos se usan para entrenar
    2. Se calcula el error en esa ronda
  - Se reporta el error promediado en las *k* rondas
- El extremo es k = n, que se conoce como leave-one-out cross-validation (LOOCV)
- También se debe usar un dataset de test para la evaluación final

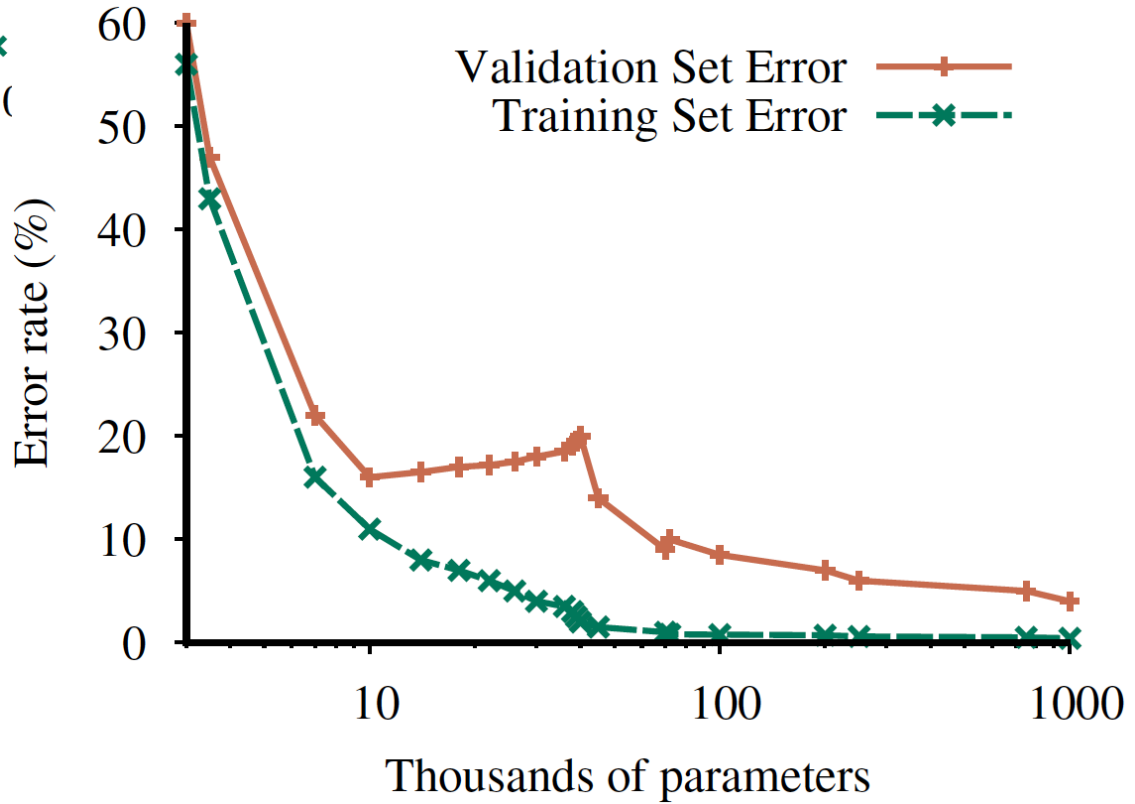
Unidad 3. Aprendizaje computacional
Validación cruzada (k-fold cross-validation)



Unidad 3. Aprendizaje computacional
Patrones típicos en selección de modelos

UNIVERSIDAD DE MURCIA





UNIVERSIDAD DE MURCIA

Unidad 3. Aprendizaje computacional

EVALUACIÓN DE MODELOS

- Clasificación, cuando la *y* es una variable categórica:
  - En clasificación binaria tenemos ejemplos positivos (pertenecen a la clase) y negativos (no pertenecen)

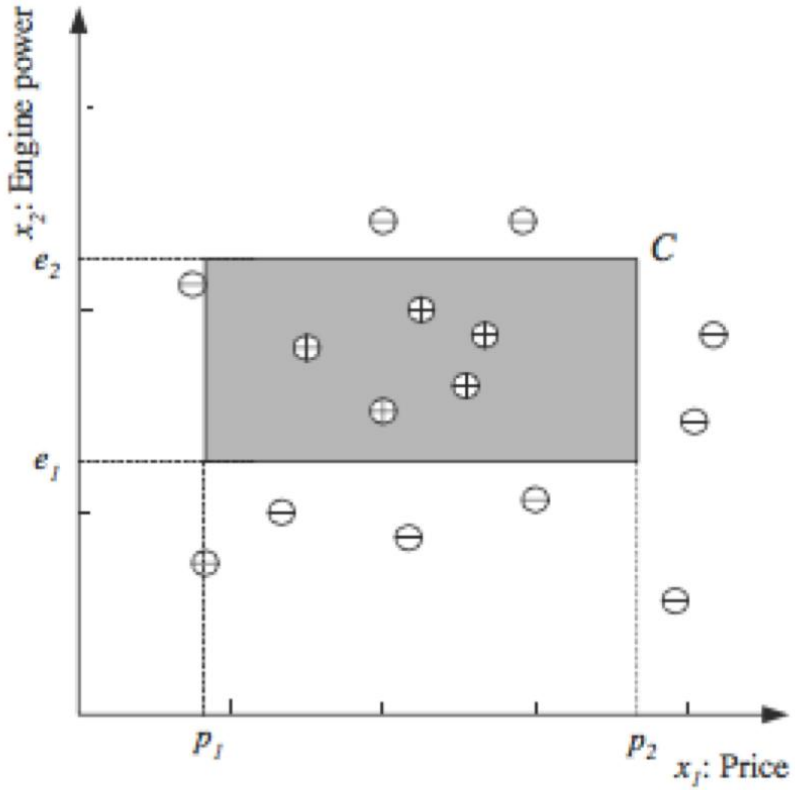


Figure 2.2 Example of a hypothesis class. The class of family car is a rectangle in the price-engine power space.

Unidad 3. Aprendizaje computacional
Evaluación: Clasificación binaria

UNIVERSIDAD DE MURCIA

- El clasificador binario predice entre dos clases posibles
- Una métrica muy intuitiva y típica es la exactitud:
  - Exactitud: grado de acierto general del modelo = $\frac{n_c}{n}$
- La matriz de confusión para dos clases

		Valor predicho		
Valor real		**CLASS +**	**CLASS -**	total
	CLASS +	TP	FN	P
	CLASS -	FP	TN	N
	total			TOT

Clasificador predice clase + pero en realidad es clase -

Los aciertos: TP+TN
- TP: true positives
- TN: true negatives

Los fallos: FP+FN
- FP: false positives
- FN: false negatives

- Veamos cómo se calcula con un ejemplo, en el que se prediga si se sale a navegar o no un barco pesquero
- Salir a navegar es la clase positiva (Si) y no salir es la clase negativa (No)
- **NO** se diferencia entre clases, ¿es bueno el clasificador?

		Valor predicho		
Valor real		**Si**	**No**	total
	Si	87	3	90
	No	6	4	10
	total			100

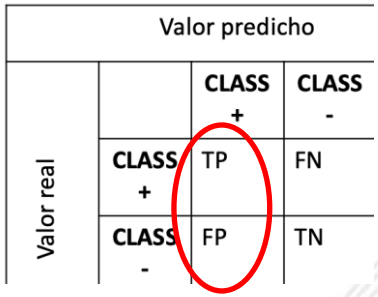
Análisis de 100 días en total
Sale 90 veces a navegar
No sale 10 veces a navegar

TP=87 (predice navegar y sale)
TN=4 (predice no navegar y no sale)
FP=6 (predice navegar y no sale)
FN=3 (predice no navegar y sale)

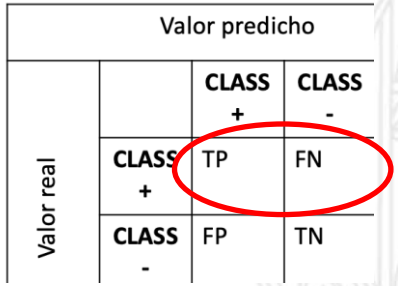
$accuracy = \frac{TP+TN}{TOT} = \frac{87+4}{100} = 0.91$, es decir exactitud 91%

- Necesitamos otras métricas para evaluar como de bien es capaz de distinguir entre las clases
- Precisión (*precision*): ¿Cómo de bueno soy prediciendo la clase positiva cuando lo hago?
- Exhaustividad o sensibilidad (recall o sensitivity): ¿Cómo de bueno es detectando la clase positiva en función de las que hay en la muestra?
- Especificidad (specificity): Como de bueno es el algoritmo detectando la clase negativa

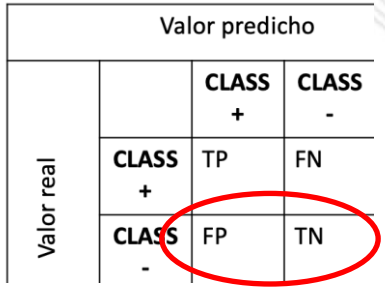
$$precision = \frac{TP}{TP + FP}$$



$$recall = \frac{TP}{TP + FN}$$



$$specificity = \frac{TN}{FP + TN}$$



Unidad 3. Aprendizaje computacional
Evaluación: Clasificación binaria

UNIVERSIDAD DE MURCIA

Valor predicho				
Valor real		**Si**	**No**	total
	Si	87	3	90
	No	6	4	10
	total			100

$$accuracy = \frac{TP + TN}{TOT} = \frac{87 + 4}{100} = 0.91$$

$$precision = \frac{TP}{TP + FP} = \frac{87}{93} = 0.93$$

$$recall = \frac{TP}{TP + FN} = \frac{87}{90} = 0.96$$

$$specificity = \frac{TN}{TN + FP} = \frac{4}{10} = 0.4$$

$$precision = \frac{TP}{TP + FP} \qquad recall = \frac{TP}{TP + FN} \qquad specificity = \frac{TN}{TN + FP}$$

Valor predicho			
Valor real		**CLASS +**	**CLASS -**
	CLASS +	TP	FN
	CLASS -	FP	TN

Valor predicho			
Valor real		**CLASS +**	**CLASS -**
	CLASS +	TP	FN
	CLASS -	FP	TN

Valor predicho			
Valor real		**CLASS +**	**CLASS -**
	CLASS +	TP	FN
	CLASS -	FP	TN

		Valor predicho		
		Si	**No**	total
Valor real	**Si**	87	3	90
	No	6	4	10
	total			100

$$F1score = \frac{2 * precision * recall}{precision + recall} = \frac{2 * 0.96 * 0.93}{0.96 + 0.93} = 0.94$$

- Clasificación, cuando la *y* es una variable categórica:
  - En clasificación multiclase tenemos múltiples clases diferentes

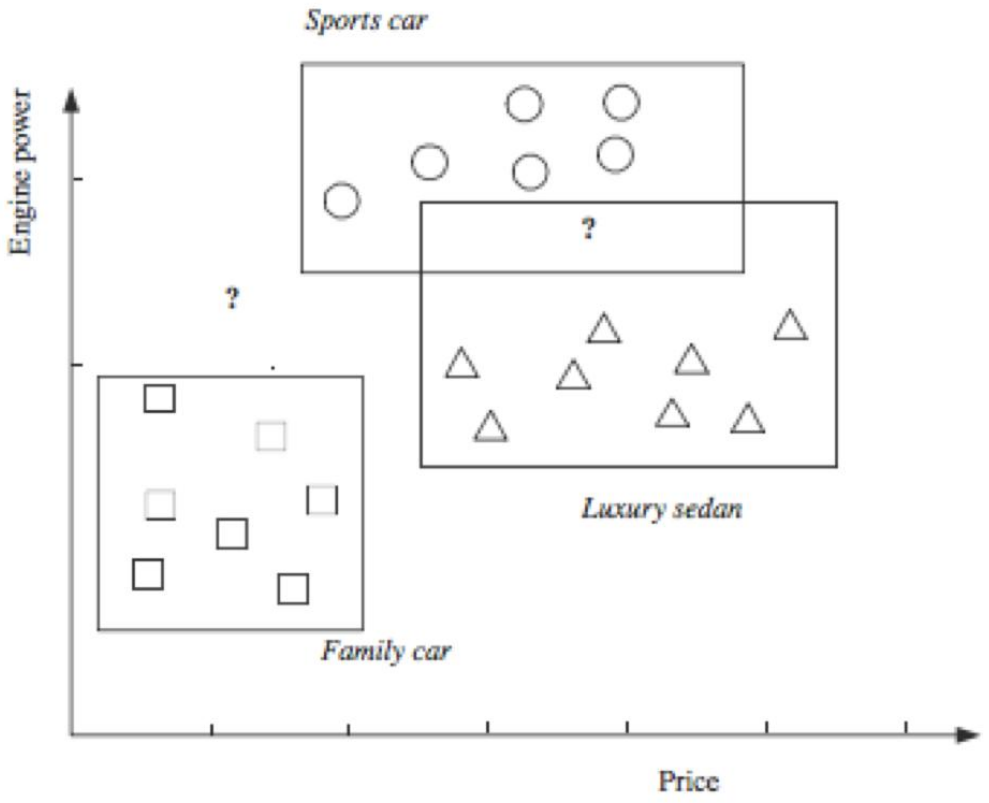
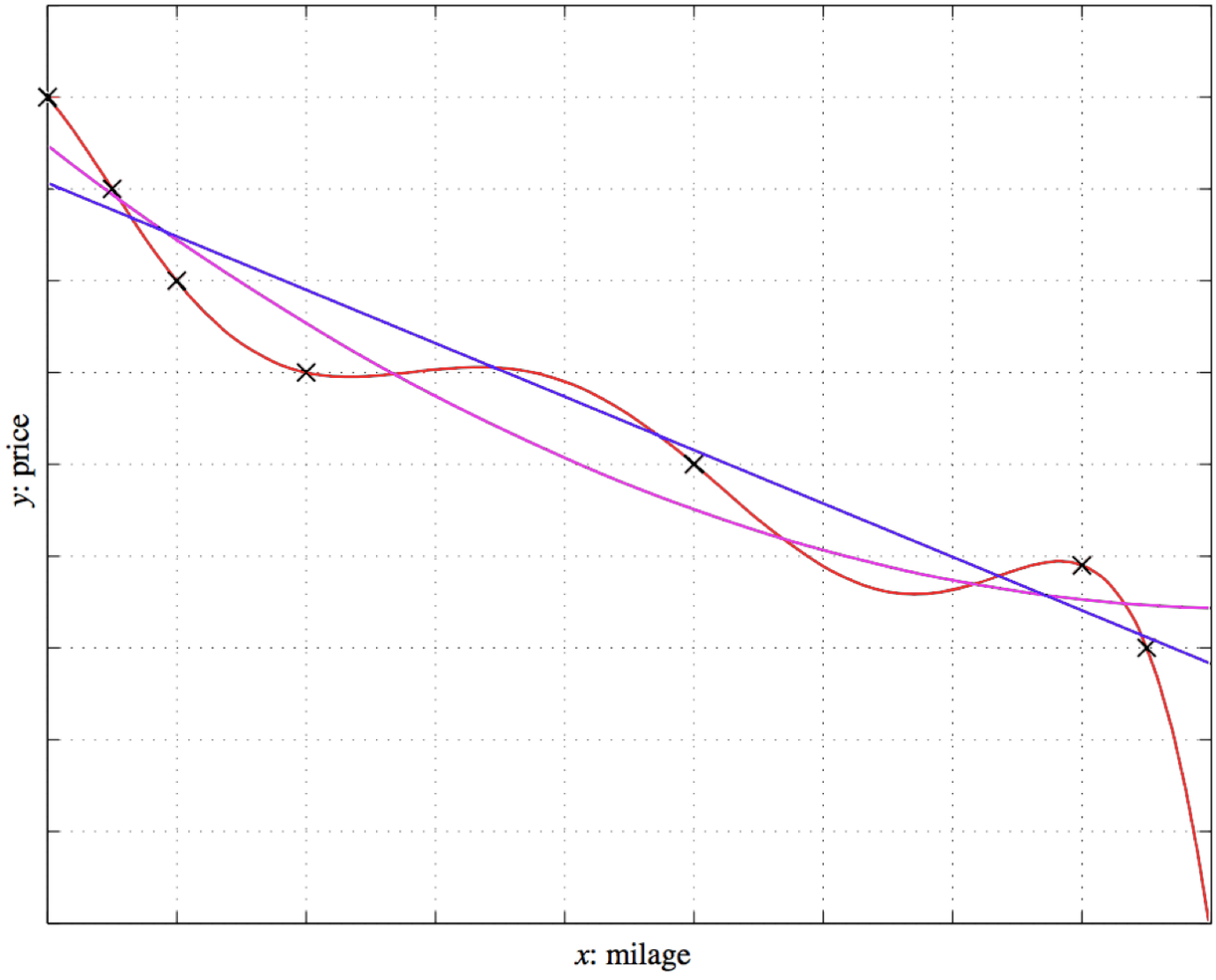


Figure 2.9 There are three classes: family car, sports car, and luxury sedan. There are three hypotheses induced, each one covering the instances of one class and leaving outside the instances of the other two classes. '?' are reject regions where no, or more than one, class is chosen.

- Se puede calcular su exactitud
- También su precisión y exhaustividad aplican, y también el F1-score
  - Sin embargo, como hay múltiples clases, estas métricas deben ser calculadas para cada clase por separado
- Por ejemplo, para el F1-score:

$$F1score = \frac{1}{K} \sum_{k=1}^{K} F1\ score\ for\ class\ k$$

- Regresión, cuando la *y* es una variable numérica:
  - Podría ser tanto un entero o número real



Tipos de problemas: Regresión

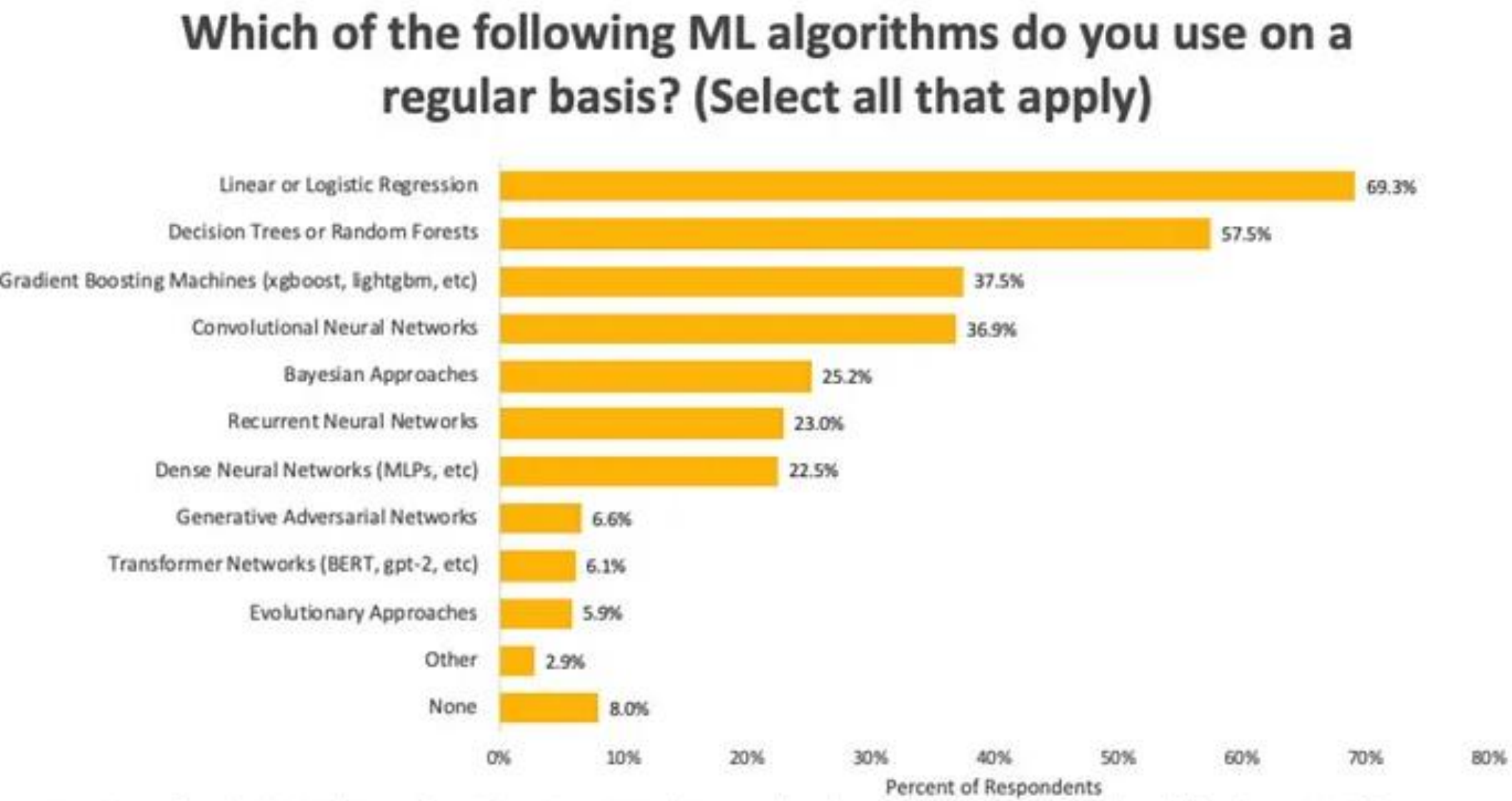
- No tenemos una clase discreta como salida, sino una predicción numérica
- Las métricas de clasificadores no aplican en este caso
- Normalmente se usa el Error Cuadrático Medio
  - Pero hay otros como la raíz cuadrada del MSE (RSME) o el error medio absoluto (MAE)
- Sea el dataset de entrenamiento $E = \{(x_1, y_1), \ldots, (x_N, y_N)\}$
- Y sea h la función hipótesis cuya salida es numérica:

$$h(x_i) = y_i^*$$

$$MSE = \frac{\sum_{i=1}^{N}(y_i^* - y_i)^2}{N}$$

Unidad 3. Aprendizaje computacional
Algoritmos más populares

UNIVERSIDAD DE MURCIA



Note: Data are from the 2019 Kaggle ML and Data Science Survey. You can learn more about the study here: https://www.kaggle.com/c/kaggle-survey-2019/data.
A total of 19717 respondents completed the survey; the percentages in the graph are based on a total of 14762 respondents who provided an answer to this question.

BUSINESS BROADWAY

UNIVERSIDAD DE MURCIA

Unidad 3. Aprendizaje computacional

MÉTODO BAYESIANOS: NAÏVE BAYES

- Son métodos usados para la clasificación utilizando la probabilidad
- Han mostrado buen rendimiento comparados con otros métodos más complejos y velocidad con grandes bases de datos
- Se basan en el teorema de Bayes como fundamento:

$$\boldsymbol{P}(\mathrm{H}\,|\,\mathbf{X})=\frac{P(\mathbf{X}\,|\,A)P(\mathrm{H})}{P(\mathbf{X})}$$

- *X* es una tupla de datos (con *n* atributos), la evidencia
- *H* es la hipótesis, que *X* pertenece a la clase *C*

- Objetivo: *P(H|X)*, la probabilidad a posteriori
  - Dado un cliente, entonces P(H|X) sería la probabilidad de que compre un ordenador si tiene tiene 35 años y salario de 40k
- P(H): Probabilidad a priori de H
  - La probabilidad de que un cliente compre un ordenador, independientemente de sus atributos
- P(X|H): Probabilidad a posteriori condicionada a H. Verosimilitud de la hipótesis dado el conjunto de observaciones
  - La probabilidad de que un cliente tenga 35 años y gane 40k, sabiendo que comprará un ordenador
- P(X): Probabilidad a priori sobre la ocurrencia
  - La probabilidad de que una persona en nuestros datos tenga 35 años y gane 40k
- Estas probabilidades son **estimables** con los datos

Unidad 3. Aprendizaje computacional
Otra interpretación

- Otra posible interpretación sería:

$$P(Causa|Efecto) = \frac{P(Efecto|Causa)P(Causa)}{P(Efecto)}$$

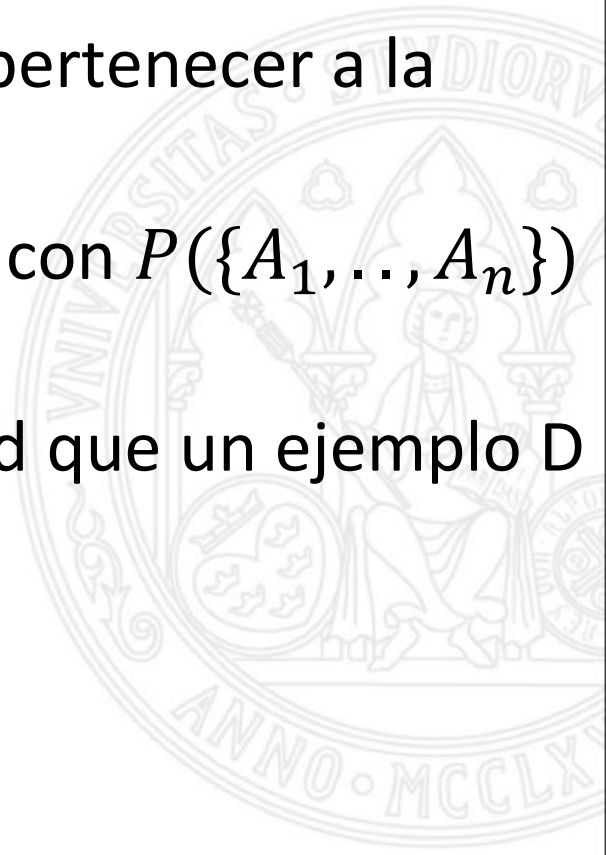
  - P(Efecto|Causa) cuantifica la relación causal
  - P(Causa|Efecto) indica la capacidad de predicción/diagnóstico
- Ejemplo: Un médico para diagnosticar, le vendría bien saber la probabilidad de tener COVID si tiene tos
  - Se analizan los registros de la clínica y sabemos:
    - El número de pacientes con tos P(tos)
    - El número de pacientes con diagnóstico COVID P(COVID)
    - La probabilidad de tener tos, dada el COVID P(tos|COVID)

Teorema de Bayes en problemas de clasificación

- En el contexto de clasificación, tendríamos:

$$P(C|\{A_1,..,A_n\}) = \frac{P(\{A_1,..,A_n\}|C)P(C)}{P(\{A_1,..,A_n\})}$$

- C ($\{C_1,\ldots,C_m\}$, la clase, P(C), la probabilidad de pertenecer a la clase C
- $\{A_1,\ldots,A_n\}$: cada caso, descrito como atributos, con $P(\{A_1,..,A_n\})$ siendo la probabilidad de que ocurra el caso
- $P(\{A_1,..,A_n\}|C)$: P(D|C) indica la probabilidad que un ejemplo D pertenezca a la clase C



- Naïve Bayes - Basado en dos fundamentos principales:
  1. Teorema de Bayes
  2. Suposición que los atributos son independientes, conocido el valor de la clase
- Esta suposición (2) es un tanto extrema y poco realista en la mayoría de problemas. Sin embargo, la investigación ha mostrado que aunque no se cumpla, el algoritmo funciona
- Naïve Bayes – funcionamiento
  1. Intenta encontrar la clase más probable a la que pertenece el caso
  2. Clase más probable (criterio MAP)
- Maximum a Posteriori (MAP): Aquella clase con la máxima probabilidad a posteriori:

$$P(C_i|X) > P(C_j|X) \text{ para } 1 \leq j \leq m, i \neq j$$

- Para obtener la hipótesis más plausible, tenemos que:

$$c_{MAP} = argmax_{c \in \Omega_m} p(c|A_1, \ldots, An) =$$

$$argmax_{c \in \Omega_m} \frac{P(A_1, \ldots., An|c)p(c)}{P(A_1,\ldots,An)} =$$

$$argmax_{c \in \Omega_m} P(A_1, \ldots., An|c)p(c) =$$

- Sin embargo, esta aproximación es complicada de resolver para calcular la distribución de probabilidad de muchas variables
  - Asumiendo la independencia de variables, tenemos que:

$$argmax_{c \in \Omega_m} \mathrm{p(c)} \prod_{i=1}^{n} p(Ai|c)$$

- Para analizar la probabilidad de la clase C_j, sólo importan las variables A_i y C_j

Unidad 3. Aprendizaje computacional
Naïve Bayes: Ejemplo completo

UNIVERSIDAD DE MURCIA

- Modelo predecir si cliente comprará un PC o no:
  - C = compra_pc {si, no}
- Con la evidencia:
  - A_1 = edad {joven, media, mayor}
  - A_2 = estudiante {si, no}
- En base al siguiente juego de datos de entrenamiento

ID	edad	estudiante	compra_pc
1	joven	no	no
2	joven	no	no
3	media	no	si
4	mayor	no	si
5	mayor	si	si
6	mayor	si	no
7	media	si	si
8	joven	no	no
9	joven	si	si
10	mayor	si	si
11	joven	si	si
12	media	no	si
13	media	si	si
14	mayor	no	no

Naïve Bayes: Ejemplo completo

$$argmax_{c \in \Omega_m} \mathrm{p(c)} \prod_{i=1}^{n} p(Ai|c)$$

- P(c)
  - P(compra_pc=si) = 9/14 = 0.643
  - P(compra_pc=no) = 5/14 = 0.357
- $p(Ai|c)$ = p({edad, estudiante}|c)
  - $p(edad|c)$
    - P(edad=joven | compra_pc=si) = 2/9 = 0.222
    - P(edad=joven | compra_pc=no) = 3/5 = 0.6
    - P(edad=media | compra_pc=si) = 4/9 = 0.44
    - P(edad=media | compra_pc=no) = 0/5 = 0
    - P(edad=mayor | compra_pc=si) = 3/9 = 0.33
    - P(edad=mayor | compra_pc=no) = 2/5 = 0.4

Naïve Bayes: Ejemplo completo

- $p(estudiante|c)$
  - P(estudiante=si | compra_pc=si) = 6/9 = 0.66
  - P(estudiante =si | compra_pc=no) = 1/5 = 0.2
  - P(estudiante=no | compra_pc=si) = 3/9 = 0.33
  - P(estudiante=no | compra_pc=no) = 4/5 = 0.8
- Ahora estaría terminado el entrenamiento después de haber estimado las probabilidades marginales y a priori
- Ahora podríamos predecir en base a estos parámetros nuevos casos del juego de test

Unidad 3. Aprendizaje computacional
Naïve Bayes: Ejemplo completo

- Predice el siguiente nuevo caso: X=(edad=joven, estudiante=si)
- $argmax_{c \in \Omega_m} \mathrm{p(c)} \prod_{i=1}^{n} p(Ai|c)$
- $\mathrm{p}(\mathrm{compra}_{pc} = si)\, p(edad = joven|compra_{pc} = si)\, p(estudiante = si|compra_{pc} = si)$ =

  0.643 * 0.222 * 0.66 = 0.09
- $\mathrm{p}(\mathrm{compra}_{pc} = no)\, p(edad = joven|compra_{pc} = no)\, p(estudiante = si|compra_{pc} = no)$ =

  0.357 * 0.6 * 0.2 = 0.04

- $argmax_{c \in \Omega_m} \mathrm{p(c)} \prod_{i=1}^{n} p(Ai|c)$ = compra_pc=si

UNIVERSIDAD DE MURCIA

Unidad 3. Aprendizaje computacional

MÉTODOS BASADOS EN CASOS Y VECINDAD: K-VECINOS MÁS PRÓXIMOS

- Se basan fundamentalmente en la utilización de ejemplos "vecinos" al dato que hay que procesar
- Por lo tanto, la distancia entre cada ejemplo y el dato será base
- Pueden ser aplicados tanto en supervisado como no supervisado
  - En el caso de supervisado, tanto para clasificación como regresión
- El concepto de similitud entre casos va a ser operacionalizado en función de métricas de distancia
- También se diferencia entre métodos retardados y no retardados:
  - Retardado o perezoso (*lazy*): Retrasan la decisión de generalización hasta recibir un nuevo dato a procesar
  - Métodos no retardados o anticipativos (*eager*): Construyen el modelo antes de tener que realizar la tarea. En ese momento, se eliminan los ejemplos y se guarda sólo el modelo

- ¿Cómo saber la similitud entre dos instancias o individuos?
  - Se debe elegir una función de distancia y calcularla entre ellos
- El mismo método puede funcionar distintos en base a la métrica
- Se puede adaptar la distancia para distintos tipos de problema
- Hay más funciones de distancia que la clásica o Euclídea
  - Además, se pueden definir de forma ad-hoc en base al problema
- Las medidas de distancia tradicionales se aplican sobre dos vectores, puntos o instancias *x* e *y* de dimensión *n*

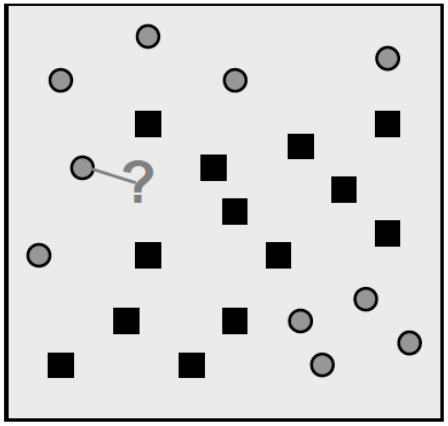
Unidad 3. Aprendizaje computacional
Distancias más usadas

UNIVERSIDAD DE MURCIA

- Distancia Euclídea: Distancia clásica, la longitud de la recta que uno dos puntos en el espacio
  - $d(x,y) = \sqrt{\sum_{i=1}^{n}(xi - yi)^2}$
- Distancia de Manhattan: Recorrer un camino no en diagonal (más corto), sino zigzagueando (como en Manhattan)
  - $d(x,y) = \sum_{i=1}^{n}|x_i - yi|$
- Distancia de Chebychev: Se calcula la discrepancia más grande de alguna de las dimensiones
  - $d(x,y) = max_{i=1\ldots n}|x_i - yi|$
- Distancia del coseno: Considerando cada ejemplo como un vector, sería el coseno del ángulo que forman
  - $d(x,y) = \frac{x*y}{\|x\|\|y\|}$

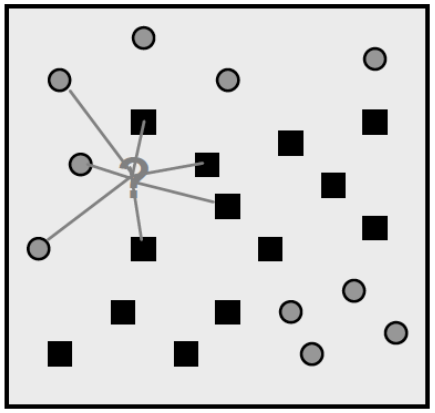
- Conveniente normalizar los atributos (entre cero y uno, por ej)
- Si no se normaliza, aquellos atributos con magnitud mucho mayor tendrán un peso desproporcionado
- La búsqueda de valores atípicos también será importante para no frustrar la normalización
- El concepto de distancia también existe en nominales
- Para los atributos nominales se suele usar la función delta
  - δ(x,y)=0 $\Leftrightarrow$ (a=b), si no δ(x,y)=1
  - Mediante esta función de distancia: $d(x,y) = w \sum_{i=1}^{n} \delta(xi,yi)$
  - Donde w es un factor reducción (1/n normalmente)
- Se puede adaptar usar una distancia para nominales y otra para numéricos, y combinar usando el factor adecuado
- Otras distancias como la de edición (inserciones, borrados…): Palabra “jamón” más cercana de “amor” que de ”vivaz”

- La regla del vecino más próximo simplemente asigna la clase del ejemplo más cercano: 1-NN (one nearest neighbor)



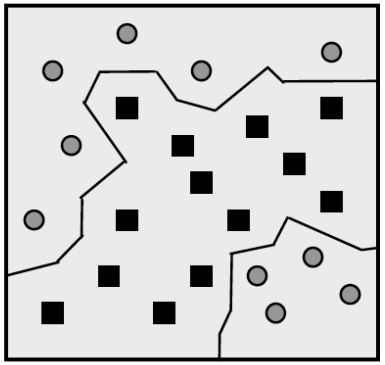
Clasifica círculo

1-nearest neighbor



Clasifica cuadrado

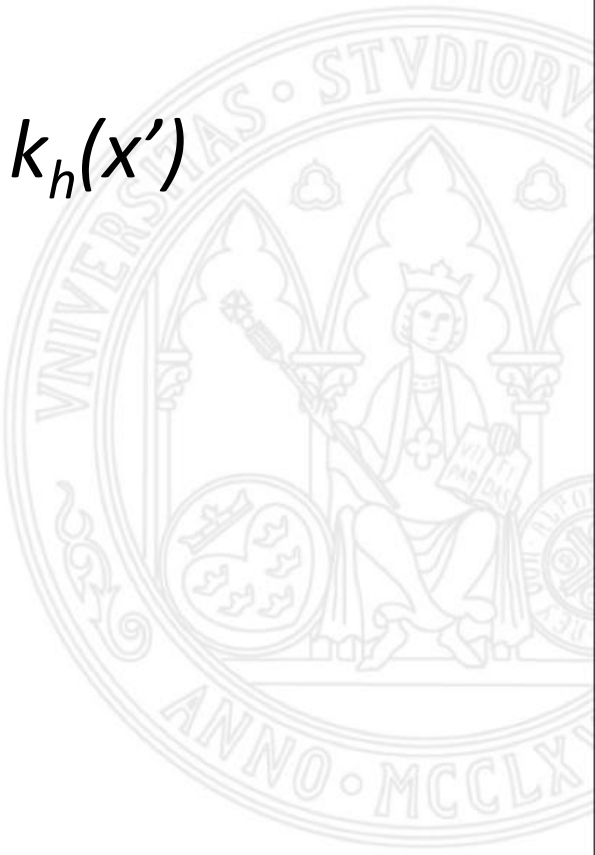
7-nearest neighbor



Partición poliédrica o de Voronoi

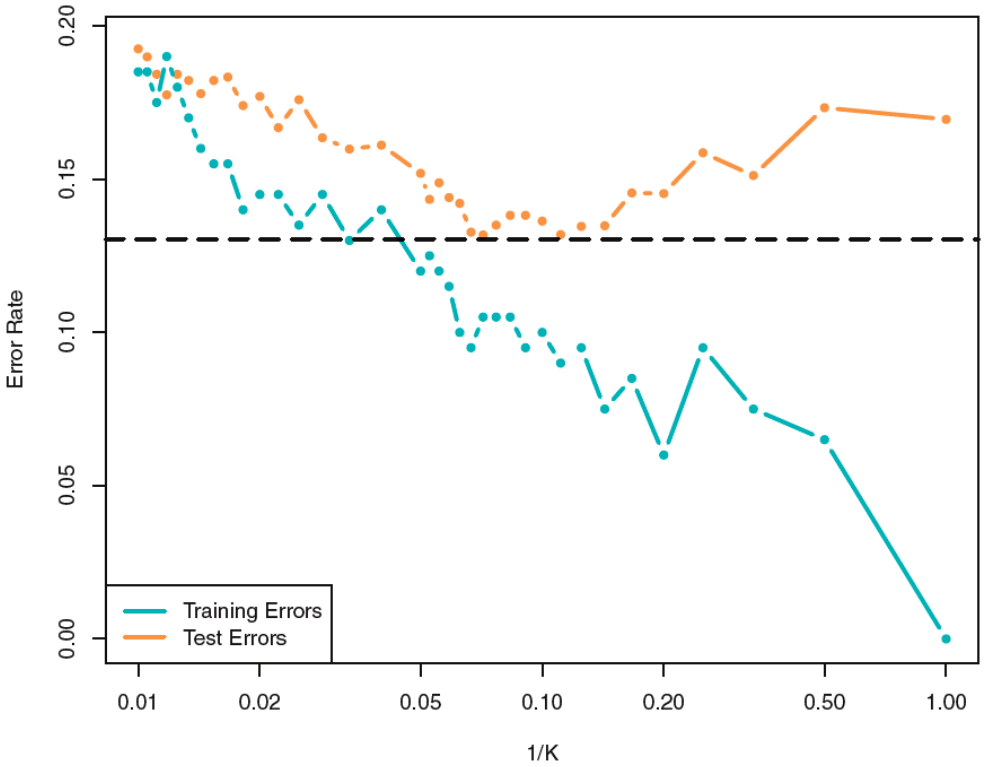
- Una variante son los k vecinos más próximos (kNN)
- K es un valor muy importante y difícil de establecer
- La expresividad del método es alta como se puede ver en la partición de Voronoi

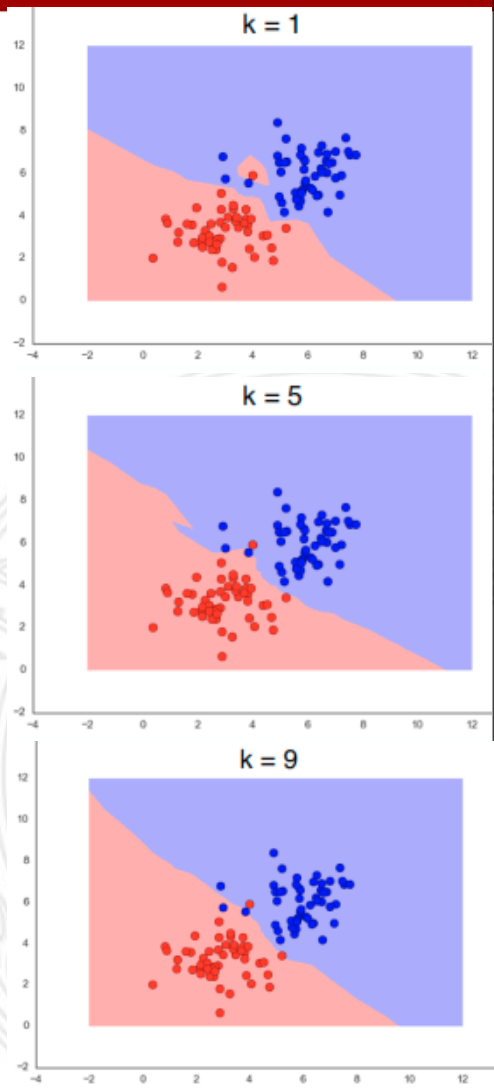
1. Dado un nuevo *x*, se genera un conjunto *x'* con los ejemplos a una distancia *r* menor de *x*
2. Se calculan los $k_h(x')$. El número de ejemplos que pertenecen a la clase A_h
3. Sea $k_g(x')$ el de mayor valor de todos los $k_h(x')$
   - $k_g = Max_{h=1\ldots q}\{k_h\}$
4. Se le asigna a *x* la clase A_g



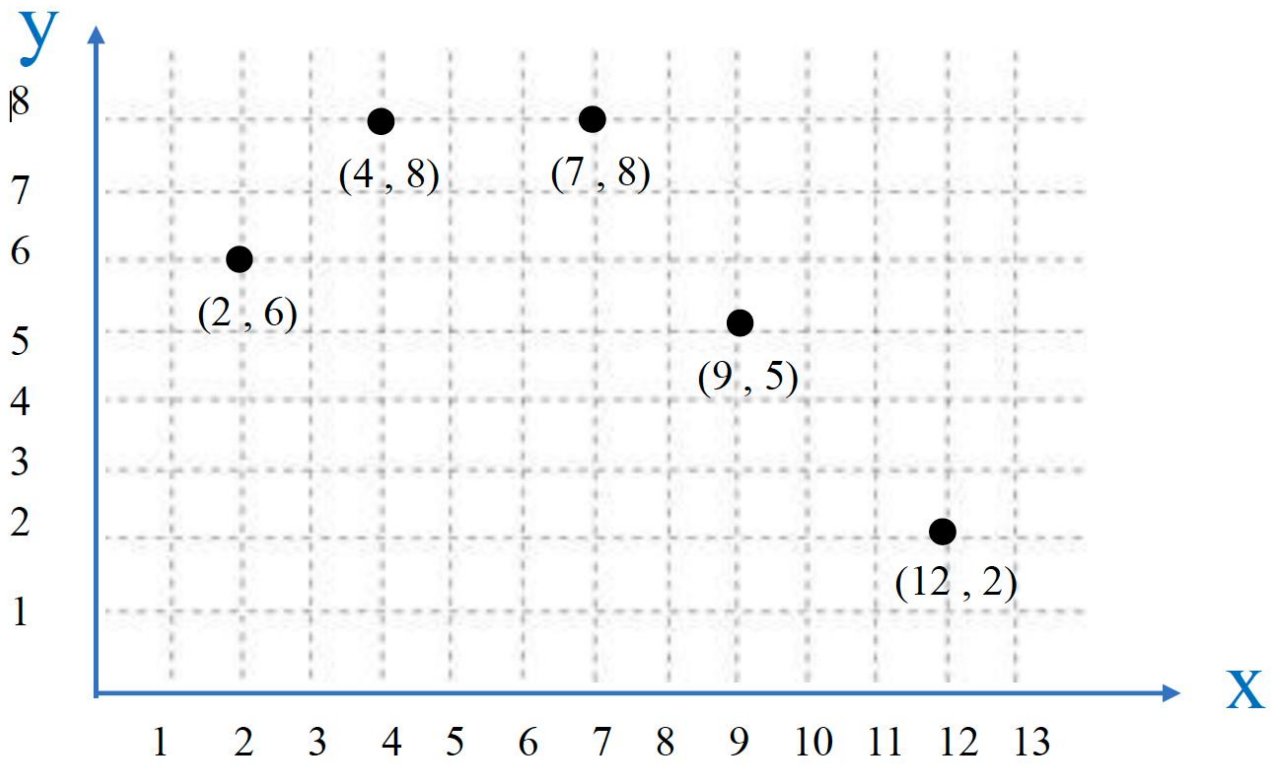
Unidad 3. Aprendizaje computacional
Elección de K: Clasificación

- K=1 es un modelo muy flexible, con poco sesgo pero alta varianza
- Conforme K crece, es menos flexible, las fronteras de decisión se vuelven más lineales: Baja varianza pero alto sesgo
- El valor óptimo de K:





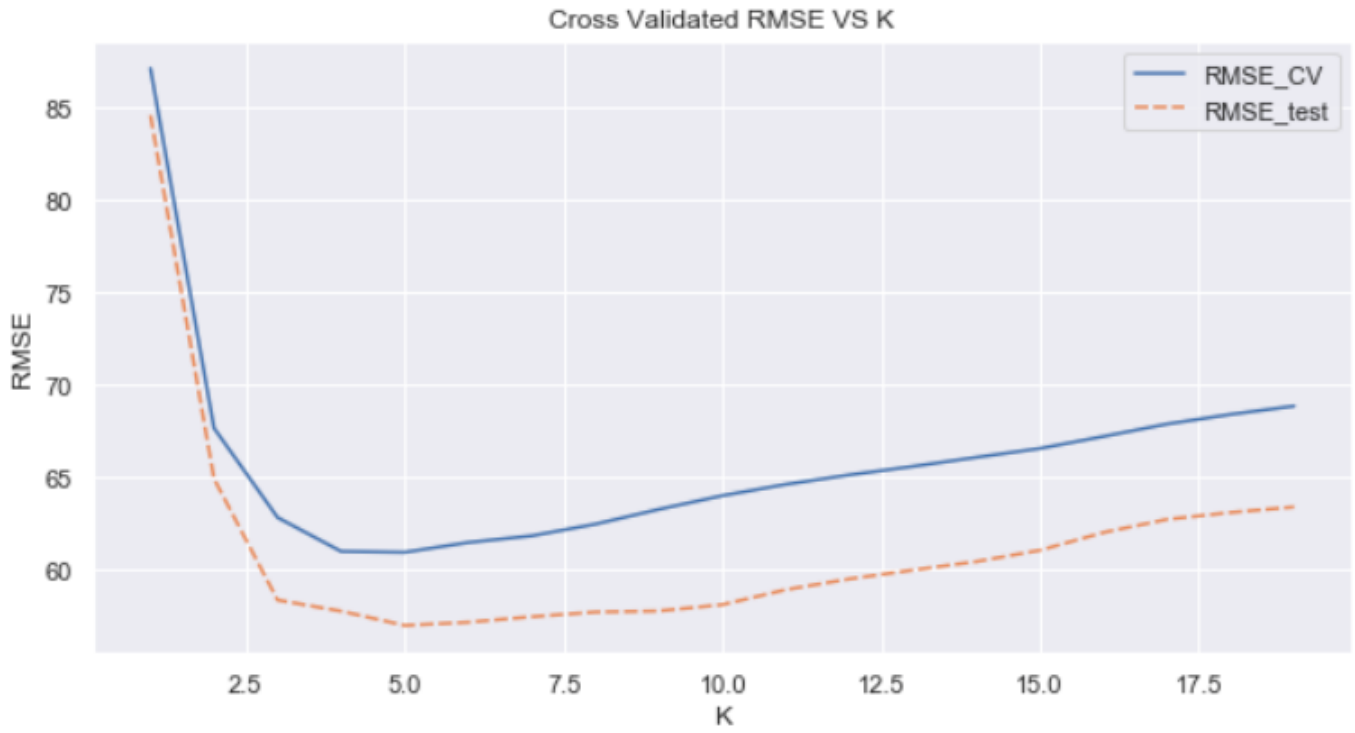
1. Dado un nuevo *x*, se genera un conjunto *x'* con los ejemplos a una distancia *r* menor de *x* que tendrá longitud k
2. Entonces $\hat{y} = \frac{1}{K} \sum_{i=0}^{k} y_i$

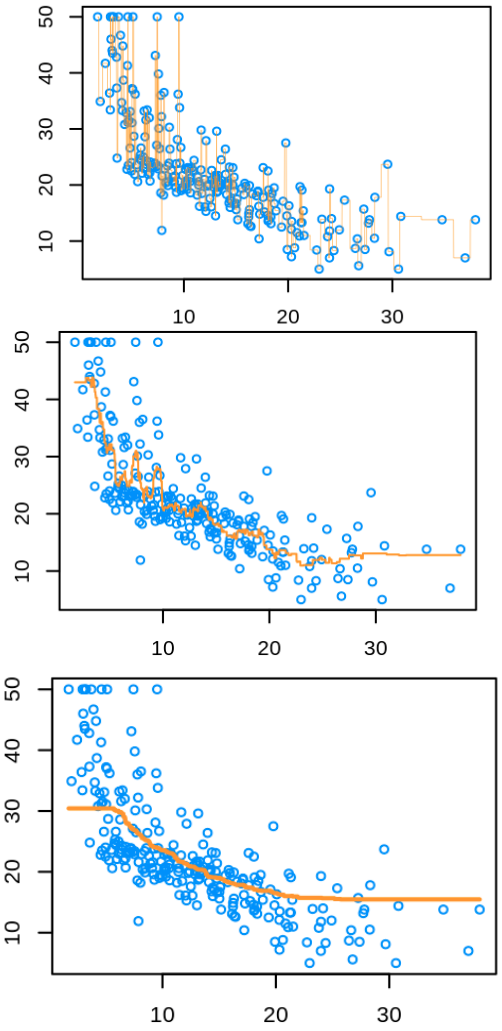


Unidad 3. Aprendizaje computacional
Elección de K: Regresión

UNIVERSIDAD DE MURCIA

- K=1 es un modelo muy flexible, con poco sesgo pero alta varianza
- Conforme K crece, es menos flexible, las fronteras de decisión se vuelven más lineales: Baja varianza pero alto sesgo
- El valor óptimo de K:





Unidad 3. Aprendizaje computacional
Beneficios y problemas

UNIVERSIDAD DE MURCIA

- Beneficios:
  - Intuitivo y simple
  - No hay asunciones
  - Fácil de implementar en problemas multi-clase
  - Usado tanto para clasificación como para regresión
  - Pocos parámetros para ajustar
- Problemas:
  - Lento (aproximación basada en memoria)
  - Maldición de la dimensionalidad
  - Funciona regular con características que son categorías
  - Elección de K
  - No hay interpretación posible