MUBD

Màster Universitari en Enginyeria de Dades Massives (Big Data)

Estadística

Índice

Statistical Model vs. Machine Learning (I)

Statistical Model

- La relación **señal/ruido es pequeña**
- No se dispone de datos de entrenamiento perfectos: **la respuesta está evaluada sin error**
- Se desea aislar los efectos de un **pequeño número de variables** como, por ejemplo, el efecto de una intervención.
- Se busca la **incertidumbre** en una predicción.
- Las **relaciones** son **aditivas** o el número de **interacciones** es **relativamente pequeño**.
- El tamaño de **muestra** es **moderado** o **pequeño**
- Se quiere un modelo **interpretable**

Statistical Model vs. Machine Learning (II)

Machine Learning

- La relación **señal/ruido es grande**
- La respuesta **NO** tiene un gran componente de **aleatoriedad**
- El objetivo es la **predicción** global sin poder describir de manera explícita el impacto de una variable determinada (p.ej, una intervención)
- **No** interesa la **incertidumbre** en las predicciones ni los **efectos** de los predictores seleccionados
- Se espera que **no** exista **aditividad**
- El tamaño de la **muestra** es **considerable** (*Big data*)
- No importa que el algoritmo sea una "**caja negra**“, si predice la respuesta

Clustering supervisado

Características

- **Se dispone de una variable respuesta**. A diferencia del clustering no supervisado, se tiene una variable respuesta categórica (nominal o ordinal) que representará los clústeres
- **Fácil evaluación de la capacidad predictiva**. Existen múltiples medidas para evaluar el rendimiento de nuestro modelo.
- **Tipo de asignación a un clúster**. Existen dos tipos de asignación a los clústeres
  - **Determinista**: cada elemento pertenece a un clúster
  - **Probabilística**: cada elemento tiene una determinada probabilidad de pertenecer a un clúster.

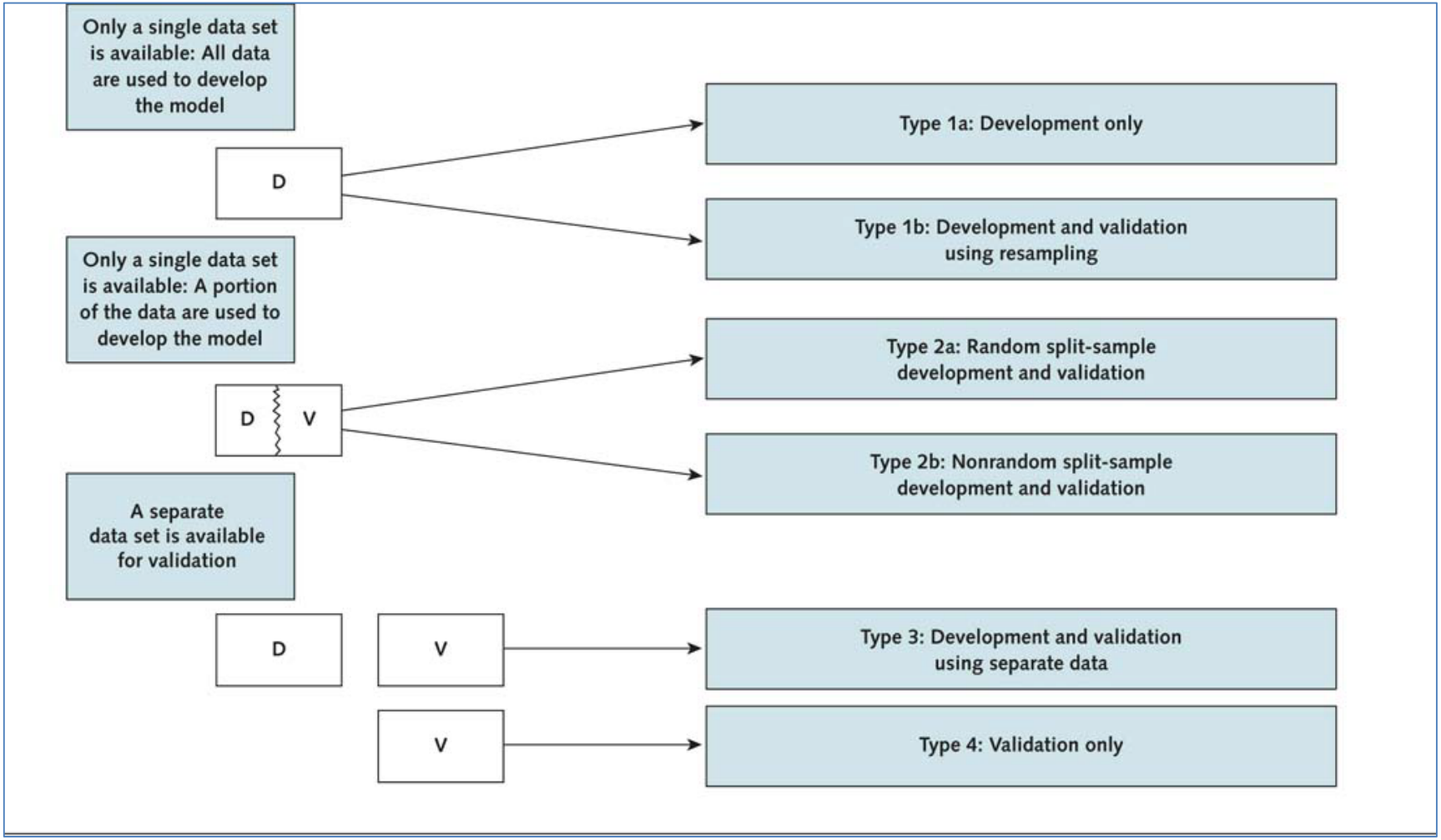
División de la muestra

Muestra de entrenamiento y prueba (I)

- Es conveniente dividir la muestra en 2 para la construcción de modelos predictivos.
- División de la muestra.
  - Muestra de Entrenamiento (TRAINING): datos con los que se construyen los modelos. Generalmente 2/3 de la muestra
  - Muestra de Prueba (TEST): datos para evaluar la capacidad predictiva de los modelos seleccionados. Generalmente 1/3 de la muestra.
- Existen alternativas de división (ver siguiente diapositiva):
  - Una de ellas es remuestrear de la misma muestra y promediar las estimaciones obtenidas bajo diferentes réplicas.

División de la muestra

Muestra de entrenamiento y prueba (II)



Source: [TRIPOD guideline]

Datos ausentes

Consideraciones

- Tipos
  - **MCAR (Missing Completely At Random)**. El hecho de que una observación esté ausente es completamente independiente de cualquier variable observada o no observada.
  - **MAR (Missing At Random)**. El hecho de que una observación esté ausente está relacionado con otras variables observadas.
  - **MNAR (Missing Not At Random)**. El hecho de que una observación esté ausente está relacionado con otras variables NO observadas o con el mismo valor ausente en sí mismo.
- Se puede intuir la distinción entre MCAR y MAR verificando si los datos ausentes se correlacionan con alguna variable observable.
- Es imposible distinguir en base a criterios estadísticos entre MAR/MCAR y MNAR y se debe usar criterios basados en la lógica (fuera de la estadística).

Datos ausentes

Soluciones para datos ausentes MCAR o MAR

Worst

- Usar los datos ausentes como una **categoría propia**. Comporta resultados totalmente sesgados.
- Imputar la **media o la mediana** en variables numéricas o la **moda** en las categóricas. Reduce la variabilidad artificialmente.

Admissible

- **LOFC**. En estudios longitudinales, aplicar la técnica Last Observation Carry Forward que consiste en arrastrar el último valor disponible.
- **Analizar los casos sin missings** ("complete case analysis").

Right

- **Imputación estratificada o por subgrupos**. Asignar según la media según los valores de un subgrupo.
- **Usar un modelo multivariante**. Usar un modelo de regresión con las variables observadas como predictoras para realizar una imputación simple.

Best

- **Imputación múltiple**. Realizar varias imputaciones para los datos ausentes (realizar réplicas de los datos) y combinar las estimaciones derivadas (ver paquete *mice* de R)

Categorización de predictores continuos

Consideraciones

- En ocasiones, se puede considerar la opción de categorizar algunos predictores continuos por razones de interpretabilidad o para evitar ciertas premisas (p.ej, linealidad).
- En general, no es una buena opción, categorizar variables continuas ya que se pierde información.
- Si la variable continua no se ajusta a las premisas del modelo se pueden usar ajustes polinómicos o *splines*.

Medidas de rendimiento

Discriminación

- Exclusivo para 2 clústeres y asignación probabilística
  - **AUC**. Probabilidad de que entre un par de individuos (uno de cada clúster), el de probabilidad predicha de pertenecer a ese clúster más alta sea el que realmente pertenece a ese clúster

- Para 2 o más clústeres y asignación probabilística
  - **Función de pérdida logarítmica**. Se considerará una buena clasificación valores bajos de la siguiente expresión:

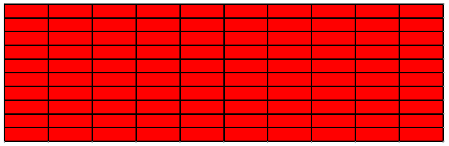
$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij}) \leftarrow \begin{cases} \textbf{N}\text{: tamaño de toda la muestra ; } \textbf{M}\text{: Número de clústeres} \\ \textbf{y}_{ij}\text{: 1 si el elemento i pertenece al clúster j y 0 en caso contrario} \\ \textbf{p}_{ij}\text{: probabilidad que el elemento i pertenezca al clúster j} \end{cases}$$

- Para 2 o más clústeres con o sin asignación probabilística
  - **Proporción de acierto**. Proporción de observaciones asignadas al clúster correcto según sus probabilidades o su asignación determinista.

Validación interna del modelo

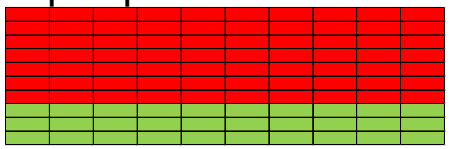
Métodos (I)

- **Rendimiento aparente**. Se usa un indicador de rendimiento que se obtiene directamente de los datos empleados para estimar el modelo.



  - Ventaja: muy simple
  - Inconveniente: da una estimación optimista (sesgada de la capacidad predictiva)
  - Ejemplo: AUC en modelo logístico sobre la muestra de entrenamiento

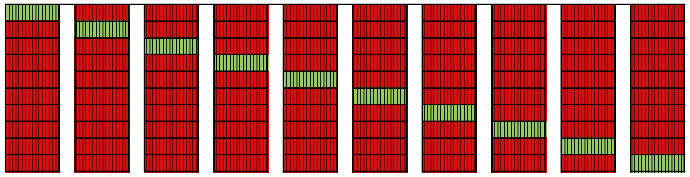
- **División de la muestra**. Se divide aleatoriamente la muestra en 2 submuestras de entrenamiento (construcción del modelo) y test (obtención del indicador de rendimiento)
  - Ventaja: simple
  - Inconvenientes: no estamos usando todos los datos para construir el modelo y suele dar también medidas del rendimiento sesgadas al alza porque la muestra se divide al azar



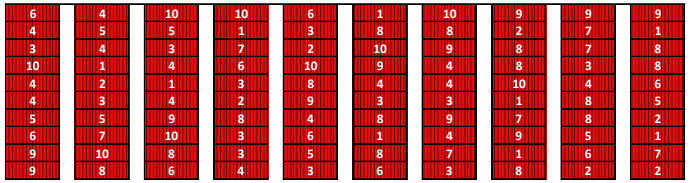
Validación interna del modelo

Métodos (II)

- **Validación cruzada**. Se divide la muestra aleatoriamente en k (p.ej. k = 10) grupos del mismo tamaño y en cada iteración se consideran *k-1* juegos como muestras de entrenamiento y 1 como muestra test. El rendimiento en una iteración se obtiene como la media de los *k-1* rendimientos y el rendimiento global es la media de todos estos.
  - Ventaja: a diferencia del anterior método, existen más probabilidades de encontrar particiones de datos “peculiares”
  - Inconveniente: requiere programación



- **Bootstrap**. Se selecciona el modelo con los datos originales. Se re-muestrea N veces con reposición sobre los mismos datos y se obtiene un modelo en cada iteración. El rendimiento de este modelo se mira sobre los datos re-muestreados y sobre los originales.
  - Ventaja: proporciona un sistema para estimar el sobreajuste comparando las dos medidas de rendimiento.
  - Inconveniente: requiere programación



Clusterización supervisado

Algoritmos y modelos

- **Modelos**
  - Regresión multinomial (respuesta categórica nominal)
  - Regresión logística (respuesta dicotómica)
  - Regresión logística ordinal (respuesta categórica ordinal)

- **Algoritmos**
  - K-Nearest Neighbors
  - Naïve bayes
  - Árboles condicionales
  - Random forest
  - Support Vector Machine

Aprendizaje basado en instancias

Introducción

- Algoritmos que comparan los nuevos elementos (instancias) con los elementos del conjunto de entrenamiento almacenados en memoria:
  - No se realizan generalizaciones, no se construyen modelos.
- La anterior propiedad implica que el cálculo computacional puede crecer considerablemente si el conjunto de entrenamiento es grande.
- El *KNN* es el ejemplo prototípico de este tipo de algoritmos de aprendizaje.

K-NN (K-Nearest Neighbors)

Introducción

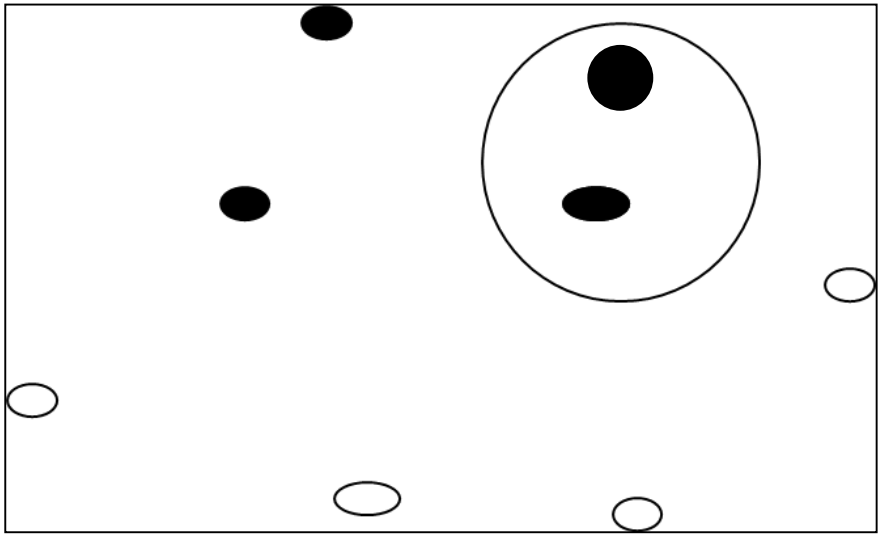
- **Definición**: método de clasificación supervisada dentro de un número determinado de clases (puede usarse con respuesta continua pero no es lo habitual)
- **Muestras**
  - **Entrenamiento**. Se dispone de una población de partida para la cuál se conocen las clases (o bien han sido asignadas con un algoritmo de clasificación no supervisada)
  - **Test**. Para cada nuevo elemento que entre en la población se busca los K-elementos más cercanos y se le asigna la clase de la mayoría de ellos.
- **¿Cuándo usarlo?**
  - Los elementos/filas se corresponden con los puntos en R^n
  - Se dispone de muchos datos de entrenamiento
  - Menos de 20 variables, generalmente estandarizadas.

K-NN (K-Nearest Neighbors)

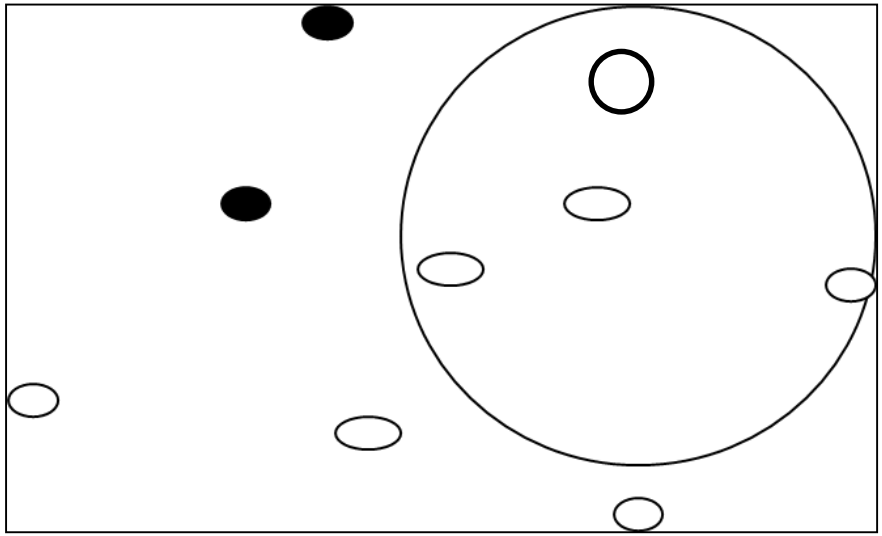
Ejemplos

- Para los puntos elípticos, se conoce la clase. El punto circular es el punto al cuál le hemos de asignar la clase.

1- Nearest Neighbor



3- Nearest Neighbor



K-NN (K-Nearest Neighbors)

Pros y contras

- Ventajas
  - Es de aplicación muy sencilla e intuitivo.
  - Gran adaptabilidad. Rehacer el "modelo" consiste en simplemente añadir o eliminar elementos de la muestra de entrenamiento.
  - No se pierde información en la fase de entrenamiento.
- Inconvenientes
  - Gran necesidad de almacenamiento del conjunto de entrenamiento
  - No proporciona información sobre los factores relevantes o irrelevantes
  - Las características irrelevantes para la clasificación proporcionan ruido que baja la eficacia del algoritmo
  - Coste computacional elevado de clasificar nuevos elementos. Pre-ordenar e indexar muestras de entrenamiento en árboles de búsqueda reduce el tiempo (*kd-tree*)
  - Aunque existen variantes para respuesta continua (p.ej., haciendo la media de los k-vecinos próximos), principalmente se usa para respuesta categórica
  - Sensible a la medida de disimilitud (distancia)

K-NN (K-Nearest Neighbors)

Algoritmo

0. Sea D = {(x_1, c_1), . . . ,(x_N , c_N)} el conjunto de puntos (x_i) para los que ya se conoce la clase (c_j) y F = {x_{N+1}, . . . ,$x_{N'}$}, los nuevos casos a clasificar

Para cada caso a clasificar desde i = ***N+1*** *hasta i =* ***N'****:*

1. Calcular la distancia $d_j = d(x_i, x_j)$ para todo objeto ya clasificado (x_j , c_j)
2. Ordenar todos los d_j en orden ascendente
3. Quedarnos con los K casos de D ya clasificados más cercanos a x_i
4. Asignar a x la clase más frecuente entre los K casos seleccionados
5. [*Opcional*: Actualizar D añadiendo el nuevo caso ya conocido]

Nota: la distancia habitual es la euclidea, pero puede usarse cualquiera.

K-NN (K-Nearest Neighbors)

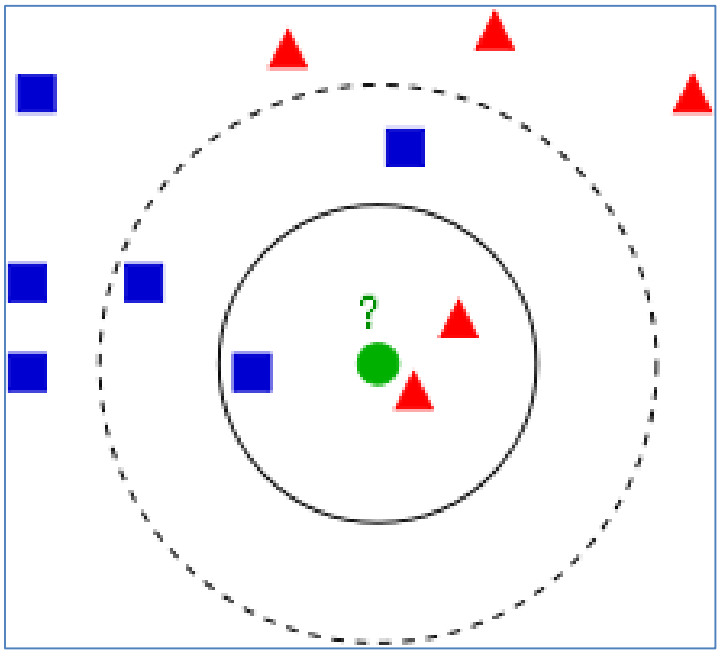
Elección de K (I)

- No hay una regla general para todos los casos
- Es una buena elección valores de K comprendidos entre 3 y 7
- Consejo: determinar experimentalmente:
  - Empezar con k = 1
  - Calcular la tasa de error con la muestra test
  - Repetir con k = k + 2 (escoger impares para evitar empates)
  - Escoger aquella k con una tasa de error menor

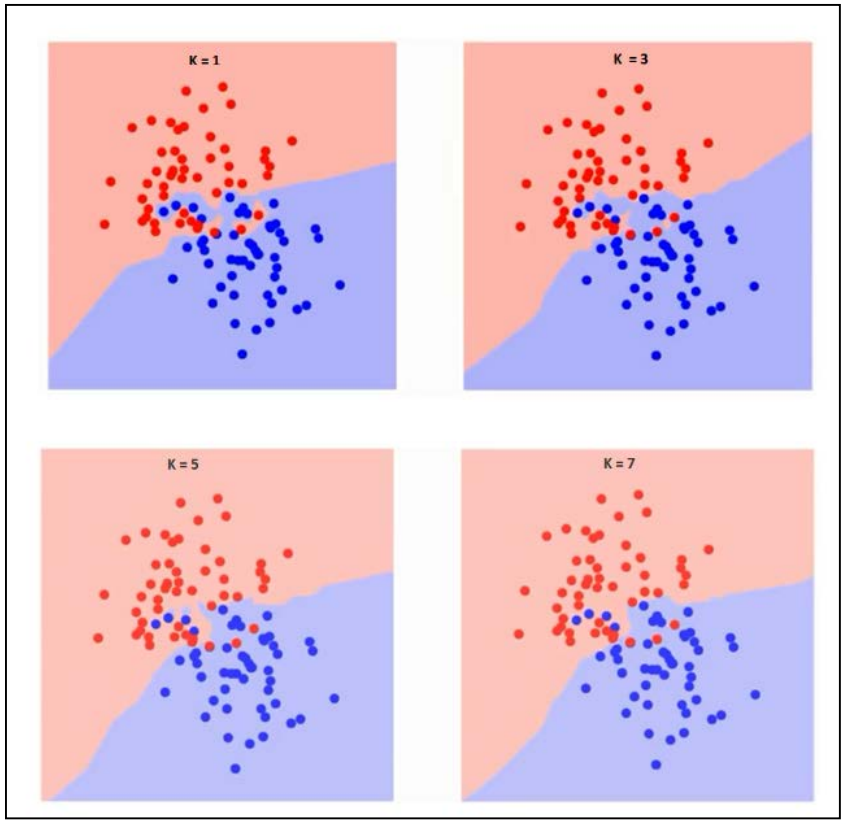
K-NN (K-Nearest Neighbors)

Elección de K (II)

- La elección de K determina el resultado
- En general K pequeñas proporcionan menos sesgo pero mayor variabilidad.



Source: Wikipedia



K-NN (K-Nearest Neighbors)

Variantes

- *K-NN con rechazo*: Sólo se asigna a una clase si un % de los K vecinos próximos pertenece a esa clase o si la diferencia entre la primera y segunda clase supera un determinado número.

- *K-NN con distancia media*: Se asigna a la clase cuya distancia media sea menor.

- *K-NN con distancia mínima*: Se escoge únicamente un representante por clase que puede ser el baricentro de todos los de esa clase. Tiene la ventaja de ser mucho menos costoso computacionalmente.

- *K-NN ponderando por los casos*: Se otorga pesos a los casos y se utiliza dichos pesos en el recuento del número de clases dentro de los k vecinos más cercanos.

- *K-NN ponderando por las variables*: Se otorga peso a las variables en el cálculo de distancia.

K-NN (K-Nearest Neighbors)

Variables irrelevantes

- Se pueden tener muchas variables pero sólo algunas de ellas ser relevantes para la clasificación
- El KNN no considera este hecho: el vecino más cercano puede ser ineficiente si hay muchas variables irrelevantes
- Soluciones:
  - Ponderar las variables de tal forma que minimicen el error de predicción
  - Usar las componentes principales resultantes de un ACP

KK-NN (KK-Nearest Neighbors)

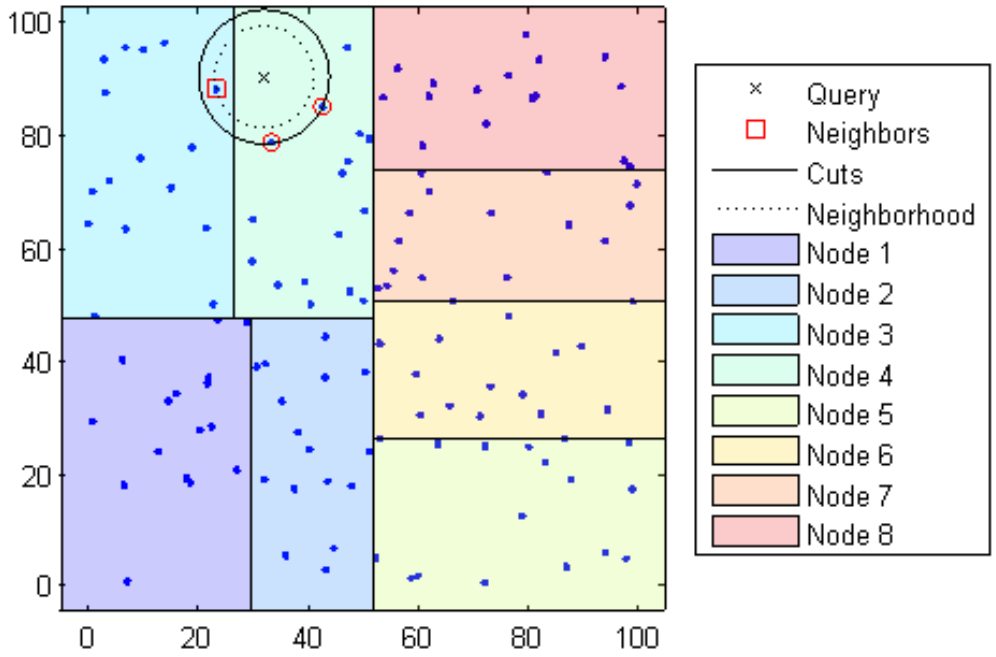
Kernel K-Nearest neighbors

- **Teorema**: Dado un conjunto de datos de entrenamiento no es linealmente separable, se puede transformar con alta probabilidad en un conjunto linealmente separable proyectándolo en un espacio de mayor dimensión a través de alguna transformación no lineal.
- **Funcionamiento**: Los nuevos elementos del conjunto test se tratan mediante un función de similitud (*Kernel*) que se calcula entre la nueva entrada y todos los elementos de entrenamiento
- **Implicaciones**: generalmente proporciona mejores predicciones aun coste computacional ligeramente más alto.

K-NN (K-Nearest Neighbors)

k-d tree

- **Definición:** árbol binario para organizar una partición de puntos en un sistema k-dimensional. Cada nodo es un punto.
- **Objetivo**: Mejorar la velocidad de clasificación



K-NN (K-Nearest Neighbors)

R

- ***knn*** {package: *class*}.
  - ***train***: datos de entrenamiento
  - ***test***: datos test
  - ***cl***: respuesta del conjunto de entrenamiento
  - ***k***: número de vecinos próximos
  - ***l***: número mínimo de vecinos de una clase para asignar
  - ***prob***: si TRUE, la proporción de votos para la clase ganadora
  - ***use.all***: si TRUE, todas las distancias máximas empatadas dentro de las k más próximas son utilizadas. Si FALSE, se utiliza una selección aleatoria para obtener

- ***CoreModel*** {package: *CORElearn*}. No solo sirve para KNN sino para otros algoritmos. No permite escoger el número de vecinos pero se complementa con el paquete *ExplainPrediction.*

- ***kknn*** {package: *kknn*}. Kernel – KNN.

- Train {package: caret}. Implementa un gran abanico de funciones.

Naive Bayes

Introducción

- **Definición**: Método de clasificación supervisada dentro de un número determinado de clases
- **Premisa**: las características que determinan la presencia o ausencia de una determinada clase son independientes entre sí.
- **Fundamento**: Se basa en el teorema de Bayes. Sean A y B dos sucesos no nulos, se cumple que:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)}$$

- **Aplicación**: Aplicado a un ejemplo de clustering supervisado:

$$P(Clase|Características) = \frac{P(Características|Clase) \cdot P(Clase)}{P(Características)}$$

Premisa de independencia

Implicaciones

- La premisa de independencia implica que cada característica influye de forma autónoma en determinar la clase aunque existan correlaciones entre dichas características.
- Formalmente sea C_j una clase correspondiente a la variable respuesta y x_1, x_2, …, x_n las características de un individuo concreto. Se tiene:

$$P(C_j|x_1, x_2, \dots, x_n) = \frac{P(x_1 \cap x_2 \cap \cdots \cap x_n \cap C_j)}{P(x_1, x_2, \dots, x_n)} = \frac{P(x_1|x_2 \cap \cdots \cap x_n \cap C_j) \cdot P(x_2 \cap \cdots \cap x_n \cap C_j)}{P(x_1, x_2, \dots, x_n)} =$$

$$= \frac{P(x_1|x_2 \cap \cdots \cap x_n \cap C_j) \cdot P(x_2|x_3 \cap \cdots \cap x_n \cap C_j) \cdot P(x_3 \cap \cdots \cap x_n \cap C_j)}{P(x_1, x_2, \dots, x_n)} = \cdots =$$

$$= \frac{P(x_1|x_2 \cap \cdots \cap x_n \cap C_j) \cdot P(x_2|x_3 \cap \cdots \cap x_n \cap C_j) \cdots P(x_n|C_j) \cdot P(C_j)}{P(x_1, x_2, \dots, x_n)} = (independencia)$$

$$= \frac{P(x_1|C_j) \cdot P(x_2|C_j) \cdots P(x_n|C_j) \cdot P(C_j)}{P(x_1, x_2, \dots, x_n)}$$

Naive Bayes

Algoritmo

- Se calcularán las probabilidades a posteriori de cada clase para cada individuo con la fórmula de Bayes asumiendo independencia entre las características:

$$P(Clase\ 1|Individuo\ 1) = 0.01$$

$$P(Clase\ 2|Individuo\ 1) = 0.1$$

$$\dots$$

$$P(Clase\ k|Individuo\ 1) = 0.2$$

- Se asignará el individuo a la categoría con mayor probabilidad (normalmente se omite el denominador porque es igual para un mismo individuo.

$$Individuo\ 1 \rightarrow Clase\ k$$

- Se puede calcular el **Factor de Bayes (FB)** entre dos categorías que nos dice cuánto más probable es una categoría respecto a otra

$$FB = \frac{P(Clase\ k|Individuo\ 1)}{P(Clase\ 2|Individuo\ 1)} = \frac{0.2}{0.1} = 2$$

- El individuo 1 tiene el doble de posibilidades de pertenecer a la categoría *K* que a la categoría 2

Variables continuas

Manejo

- Cuando se dispone de variables predictoras continuas, la probabilidad condicionada se calcula usando una distribución probabilística.
- Usualmente se utiliza la distribución Normal.
  - La probabilidad empleada será el valor de la función de densidad de la Normal (Con R → `dnorm(x,mu,sigma)`).
- Si una variable no se ajusta a la Normalidad se puede:
  - Transformar la variable
  - Emplear otras distribuciones
  - Emplear una densidad Kernel
  - Con grandes volúmenes de datos y conjunto de valores limitado, se puede considerar categórica.

Probabilidades nulas

Manejo

- El algoritmo de Naïve Bayes requiere que cada probabilidad condicional sea distinta de cero, ya que en caso contrario, la probabilidad predicha será nula
- Soluciones posibles:
  - Corrección de Laplace. Añadir un número pequeño a las frecuencias observadas (P.ej., se puede añadir 1 en numerador y denominador)
  - Añadir correcciones según las probabilidades de la categoría.
- Estas correcciones no distorsionan los resultados si la muestra es suficientemente grande

Naive Bayes

Ejemplo

Calcular el Factor de Bayes (FB) de "NO comprar" para una mujer de 30 años

Género	Edad	Compra
Hombre	40	**Sí**
Hombre	42	**Sí**
Hombre	72	**Sí**
Mujer	53	**Sí**
Hombre	41	**No**
Mujer	73	**No**
Mujer	56	**No**
Mujer	64	**No**
Mujer	43	**No**
Mujer	23	**No**

$$P(\text{Sí}|M\ y\ 30) = \frac{P(\text{Sí}) \cdot P(M|\text{Sí}) \cdot P(30|\text{Sí})}{P(M\ y\ 30)} = \frac{0.4 \cdot 0.25 \cdot 0.009}{P(M\ y\ 30)} = \frac{0.0009}{P(M\ y\ 30)}$$

$$P(No|M\ y\ 30) = \frac{P(No) \cdot P(M|No) \cdot P(30|No)}{P(M\ y\ 30)} = \frac{0.6 \cdot 0.8 \cdot 0.012}{P(M\ y\ 30)} = \frac{0.00576}{P(M\ y\ 30)}$$

$$FB = \frac{0.00576}{0.0009} = 6.4$$

Tiene 6.4 veces más posibilidades de NO comprar que de comprar

NOTA: Para los que NO compran, podemos asumir una edad Normal de media 50 y desviación 18. La densidad de 30 años en esta distribución es 0.012 [`dnorm(30,50,18)`]

Naive Bayes

Consideraciones

- **Muestras**
  - **Entrenamiento**. Servirá para calcular las probabilidades de cada característica por separado dentro de cada clase.
  - **Test**. Servirá para medir la capacidad predictiva del modelo.
- **Fórmula de Bayes**
  - El **numerador** podría expresar la *P(Características|Clase)·P(Clase)* pero para calcular el primer término se necesitarían conjuntos de entrenamiento muy grandes. De ahí, el hecho de asumir la premisa de independencia.
  - El **denominador** de la fórmula de Bayes es irrelevante ya que es constante para todas las clases, se puede prescindir de su cálculo y normalizar las probabilidades posteriormente.
- **¿Cuándo usarlo?**
  - Variables razonablemente independientes
  - Necesidad de gran velocidad computacional

Naive Bayes

Pros y contras

Ventajas:

- Aplicación muy sencilla
- Gran rendimiento computacional. Lineal respecto al número de variables predictoras. No es iterativo.
- Gran capacidad predictiva en la clasificación determinista (asignación a una clase)
- Fácilmente escalable. El incorporar una nueva característica comporta un cálculo trivial
- Conjunto de entrenamiento más pequeño que otros algoritmos

Inconvenientes:

- La premisa de independencia no siempre es cierta
- Discutible capacidad predictiva en clasificación probabilística (ya que la premisa de independencia suele no ser cierta)
- Para muestras pequeñas, $P(Características \mid Clase)$ puede ser 0 si una característica no está presente en una clase (se puede corregir este inconveniente asignando un mínimo)
- Depende fuertemente de las probabilidades a priori (probabilidades de cada clase), con lo que una muestreo no aleatorio puede sesgar nuestros resultados considerablemente

Naive Bayes

Tasa de error

- Este clasificador cogerá la clase *j* que tiene una mayor probabilidad condicionada dadas unas condiciones *X*:

$$1 - \max_j P(Y = j|X)$$

- Por tanto, este algoritmo tiene una tasa de error promedio que viene dada por la expresión:

$$1 - E\left[\max_j P(Y = j|X)\right]$$

- Esta tasa de error se puede comparar con la del algoritmo KNN. La tasa de error esperada en la clasificación con el KNN es aproximadamente el doble que con Naive Bayes bajo el supuesto de independencia.

Naive Bayes

R

- ***naiveBayes*** {package: *e1071*}.
  - ***x****: data.frame o matriz con variable numéricas y/o categóricas*
  - ***y****: Respuesta*
  - ***laplace****: corrección aplicada en variables categóricas*
  - ***subset****: subconjunto para la muestra de entrenamiento*
  - ***na.action****: ¿Qué hacer con los valores ausentes?*
- ***naive_bayes*** {package: *naivebayes*}.
- ***CoreModel*** {package: *CORElearn*}. Sirve para naive bayes y otros algoritmos.

Naive Bayes

Aplicaciones

- Clasificación de SPAM
- Reconocimiento de caracteres
- Diagnóstico
- Clasificación de productos

Naive Bayes vs. KNN

	KNN	Naive Bayes	Observaciones
Tiempo computacional	✗	✓	NB mejor rendimiento, sobre todo en fase de validación
Número de variables	✗	✓	NB puede manejar un mayor número de atributos.
Independencia	✓	✗	KNN no requiere variables no correlacionadas
Interacciones	✓	✗	NB no tiene en cuenta las interacciones
Sesgo	✓	✗	KNN proporciona estimaciones menos sesgadas
Varianza	✗	✓	NB proporciona estimaciones menos variables
Error medio	✗	✓	El error medio esperado en NB es menor
Regiones de decisión	✓	✗	KNN no depende de las formas de las regiones de decisión
Atributos irrelevantes	✗	✓	En NB, los atributos irrelevantes no contribuyen a tomar una decisión.
Interpretabilidad	✗	✓	NB no es muy interpretable, pero permite explorar el porqué de una decisión.
Parámetros	✗	✓	En KNN se debe decidir el número de vecinos
Datos ausentes	✗	✓	KNN requiere todos los valores para calcular distancias. NB, no los requiere todos para calcular probabilidades.

Bagging and Boosting

Definición y consideraciones

- **Bagging**. Agregación de predicciones mediante bootstrap: se generan muestras distintas con selección con remplazamiento de la muestra original. Objetivo: **disminuir varianza**.
- **Boosting**. Algoritmos que combinan diferentes predicciones débiles en una única predicción fuerte combinándolas de forma óptima. Objetivo: **disminuir sesgo**.
- **Consideraciones sobre estas técnicas**
  - Puede mejorar la varianza y el sesgo.
  - Evita el sobre-ajuste
  - Ambos tiene sentido usarlos cuando la capacidad predictiva inicialmente es pobre

MUBD

Màster Universitari en Enginyeria de Dades Massives (Big Data)

Estadística