TDVI: Inteligencia Artificial

Bayes ingenuo / Vecinos más cercanos

UTDT - LTD

Estructura de la clase

- Motivación
- Bayes ingenuo
- Vecinos más cercanos

Estructura de la clase

- Motivación
- Bayes ingenuo
- Vecinos más cercanos

Motivación

Diferentes algoritmos de aprendizaje pueden pertencer a diferentes familias/categorías. Una primera división:

- Modelos paramétricos. Asumen una forma funcional específica y predeterminada. Tienen un número fijo y finito de parámetros a estimar (que no varía según el tamaño del dataset). Por ejemplo: regresión lógística, regresión lineal/lasso/ridge, support vector classifier (i.e., SVM con kernel lineal)

- Modelos no paramétricos. No asumen una forma funcional específica. La pueden variar (se aprende) a medida que más complejo es el dataset de entrenamiento. Típicamente no tienen un número fijo de parámetros y utilizan los mismos datos para definir su complejidad. Por ejemplo: SVM con radial kernel, vecinos más cercanos (clase de hoy)

Motivación

Otra distinción:

- Modelos discriminativos. Modelan directamente *P(y | X).* Es decir, la probabilidad condicional. Ejemplos: árboles de decisión y regresión logística

- Modelos generativos. Es una camino menos directo. Se estima *P(X, y)* o, lo que es equivalente, *P(X | y)* y *P(y)*. Sobre la base de dicha estimación, se puede utilizar el teorema de Bayes y obtener *P(y | X)*. Algo extra es que permiten hacer es generar/simular instancias de X dada una clase (e.g., generar una imagen de un gato)

En esta clase, por completitud, veremos modelos diferentes a los que hemos visto hasta ahora

Estructura de la clase

- Motivación
- Bayes ingenuo
- Vecinos más cercanos

Bayes ingenuo

Bayes ingenuo (algoritmo de clasificación). Modela la probabilidad de que una observación *i* con determinadas características (x_i) pertenezca a una determinada clase *k*, es decir: $P(\, C_k \mid x_i \,)$. Sigue una estrategia generativa

Desarrollemos la expresión:

$P(\, C_k \mid x_i) = P(\, C_k \,,\, x_i) \,/\, P(\, x_i \,)$ *por probabilidad condicional*

$P(\, C_k \mid x_i) = (\, P(\, x_i \mid C_k) * P(\, C_k \,) \,) \,/\, P(\, x_i \,)$ *Teorema de Bayes*

$P(\, C_k \mid x_i) = (\, P(\, v_1 = x_{i1} \wedge v_2 = x_{i2} \wedge \ldots\, v_q = x_{iq} \mid C_k) * P(\, C_k \,) \,) \,/\, P(\, v_1 = x_{i1} \wedge v_2 = x_{i2} \wedge \ldots\, v_q = x_{iq})$

Bayes ingenuo

$$P(C_k \mid x_i) = (P(v_1 = x_{i1} \wedge v_2 = x_{i2} \wedge \ldots v_q = x_{iq} \mid C_k) * P(C_k)) / \underbrace{P(v_1 = x_{i1} \wedge v_2 = x_{i2} \wedge \ldots v_q = x_{iq})}_{\text{Irrelevante (}C_k\text{ no interviene!)}}$$

$$P(C_k \mid x_i) \propto (\underbrace{P(v_1 = x_{i1} \wedge v_2 = x_{i2} \wedge \ldots v_q = x_{iq} \mid C_k)}_{\text{Complejo de estimar}} * P(C_k))$$

Para solucionar este problema, el algoritmo hace un supuesto "ingenuo": las variables poseen independencia condicional *(P(A, B | C) = P(A | C) * P (B | C))*

$$P(C_k \mid X_i) \propto (\underbrace{P(v_1 = x_{i1} \mid C_k) * P(v_2 = x_{i2} \mid C_k) * \ldots * P(v_q = x_{iq} \mid C_k)}_{\text{Likelihood}}) * \underbrace{P(C_k)}_{\text{Prior}})$$

Al conjunto de supuestos usados para definir un modelo de aprendizaje se lo conoce como sesgo inductivo

Bayes ingenuo

¿Cómo se pueden calculan las probabilidades condicionales para variables categóricas?

Véamoslo para Give Birth y para Can Fly

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

Bayes ingenuo

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

$$p(C_k)\prod_{j=1}^{q} p(x_{ij} \mid C_k)$$

$$P(A|M) = \frac{6}{7}\times\frac{6}{7}\times\frac{2}{7}\times\frac{2}{7} = 0.06$$

$$P(A|N) = \frac{1}{13}\times\frac{10}{13}\times\frac{3}{13}\times\frac{4}{13} = 0.0042$$

$$P(A|M)P(M) = 0.06\times\frac{7}{20} = 0.021$$

$$P(A|N)P(N) = 0.004\times\frac{13}{20} = 0.0027$$

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

*P(A | M) * P(M) > P(A | N) * P(N)* → Se predice mamífero

Bayes ingenuo

Detalles:

El algoritmo se puede adaptar a atributos continuos. Estrategias posibles:

- Discretizando atributos categóricos (como vimos en la clase de ingeniería de atributos)
- Efectuando estimaciones de funciones de densidad
  - Paramétricas: por ej. asumiendo normalidad (Tan, pp. 233)
  - No paramétricas, por ej. utilizando estimadores de densidad de *kernel* (Alpaydin, pp. 186). MUY COSTOSO (en cómputo y espacio)

Bayes ingenuo

Detalles:

¿Qué sucede en el ejemplo anterior con *P(lives in water = sometimes | mammal)*?

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

Bayes ingenuo

Detalles:

¿Qué sucede en el ejemplo anterior con *P(lives in water = sometimes | mammal)*?

La probabilidad condicional estimada es 0, entonces para dicha clase se predecirá una probabilidad estimada nula (esto no parece buena idea…)

Por esto se suele modificar la forma de estimar las probabilidades de la siguiente manera (suavizado laplaciano o aditivo):

$$P(v_j = x_{ij} | C_k) = \frac{n_{x_{jk}} + \alpha}{n_k + \alpha K}$$

Valores comunes:

K = # valores distintos de la variable x (3 para "Live in Water")

α = 1 (*add-one smoothing*, ¿Por qué?)

Bayes ingenuo

Puntos a favor:

- Simple de entender y de implementar
- Se entrena fácilmente, incluso anda bien con datasets pequeños
- Es muy rápido
- Es relativamente insensible a atributos irrelevantes
- Puede manejar de manera simple valores nulos (no lo hace *sklearn*)
- Una vez entrenado, ocupa poco espacio (salvo que se usen estimaciones no paramétricas de densidad en atributos continuos)

Puntos en contra:

- El supuesto de independencia condicional suele ser irreal (pero no necesariamente lo es)
- No suele tener gran performance (si algo anda peor que Bayes ingenuo, muy probablemente algo esté mal)
- No descubre relaciones complejas

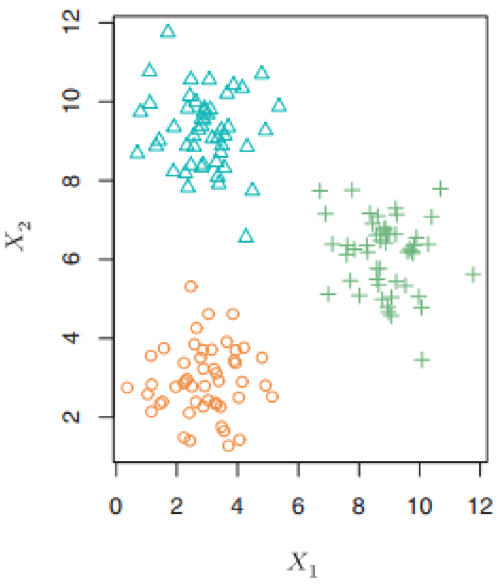
Estructura de la clase

- Motivación
- Bayes ingenuo
- Vecinos más cercanos

Vecinos más cercanos

Vecinos más cercanos (knn), es un modelo no paramétrico

Recordemos que podemos ver los registros como puntos en el espacio de atributos (*feature space*)



Propongamos alguna medida de distancia que contemple todo par de observaciones

Vecinos más cercanos

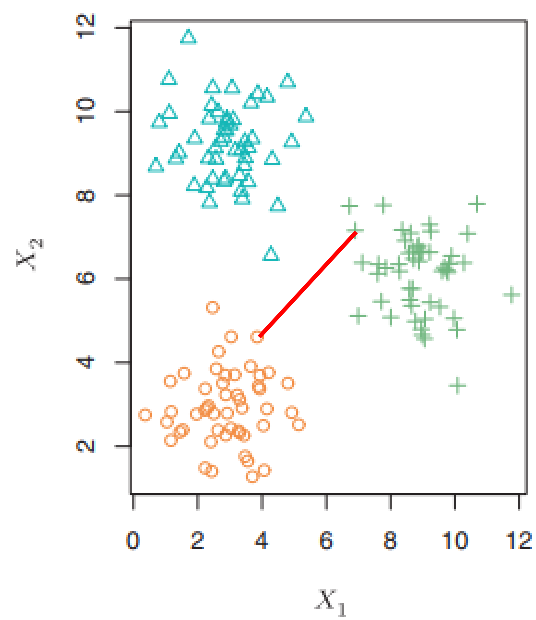
La medida de distancia más común de usar es la distancia euclídea:

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1-p_1)^2 + (q_2-p_2)^2 + \cdots + (q_n-p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2}.$$

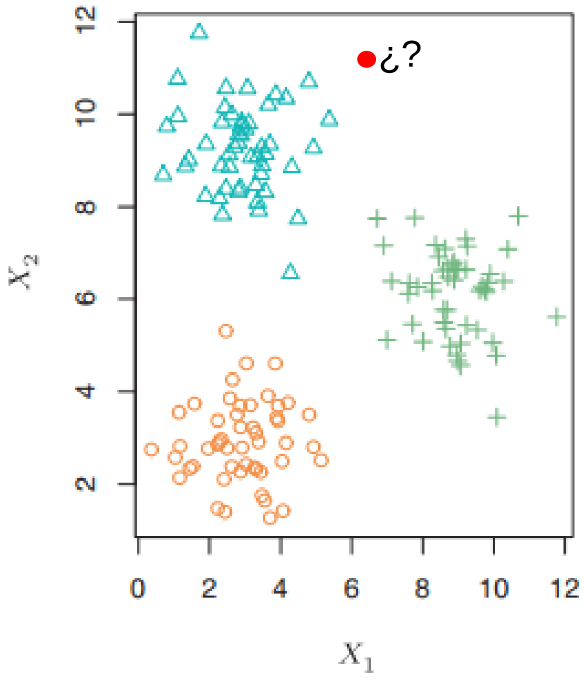
Algunas otras medidas de distancias/similitud alternativas son:

- Manhattan distance (parecida a euclidea)
- Mahalanobis distance (atributos normales)
- Cosine similarity (embeddings)
- Levenshtein distance (secuencias)
- Hamming distance (atributos categóricos)



Vecinos más cercanos

¿Supongamos que llega una nueva observación cuya clase no conocemos, qué clase parece razonable asignarle?

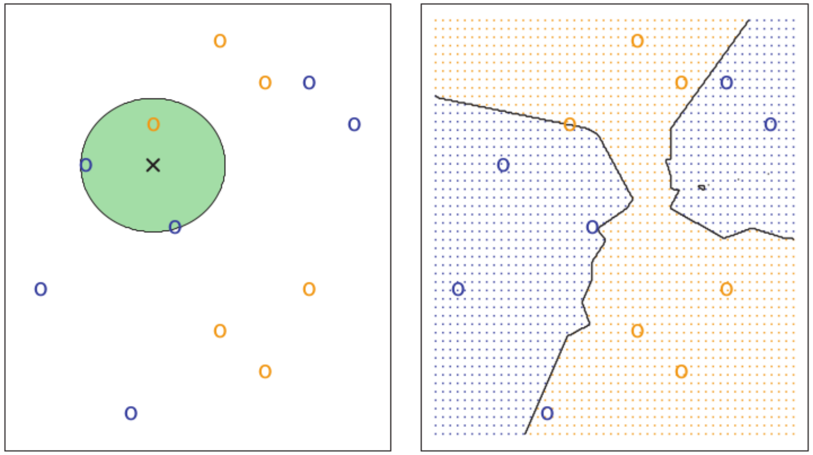


Vecinos más cercanos

Dada una observación a clasificar (x_0) y un valor de K:

1. Se identifican los k vecinos (N_0) más cercanos de x_0
2. Se estima la probabilidad condicional de pertenecer a la clase j como la fracción de observaciones de N_0 que pertenecen a j

$$\Pr(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j)$$



Vecinos más cercanos

El valor de *K tiene un efecto dramático* en el comportamiento del clasificador

- Valores de *K* pequeños hacen que la frontera de decisión sea muy flexible
- Valor de *K* grandes hacen que la frontera sea menos flexible

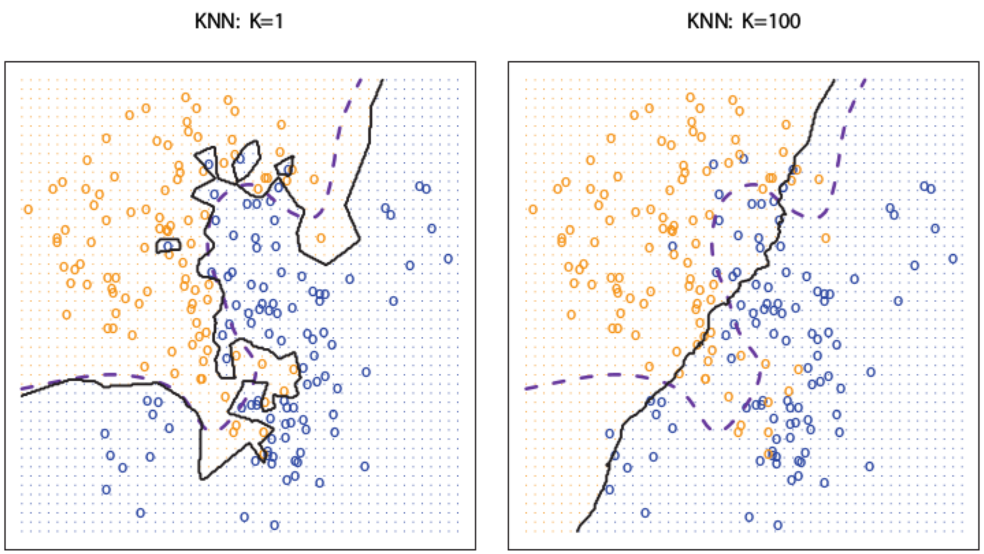
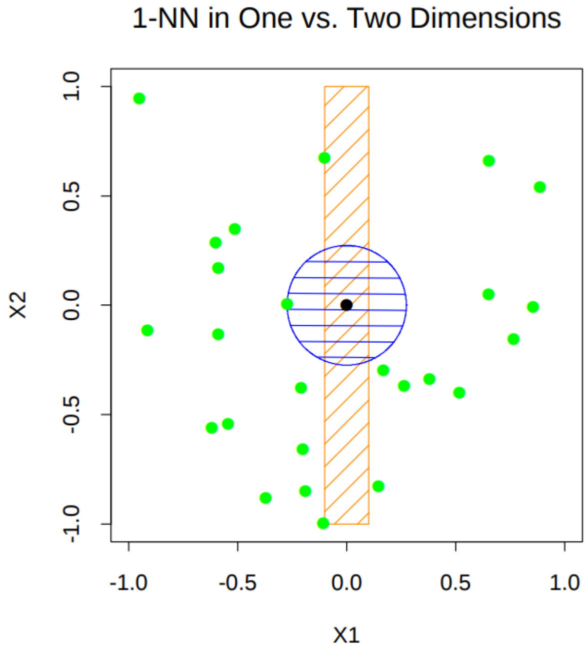


FIGURE 2.16. *A comparison of the KNN decision boundaries (solid black curves) obtained using $K = 1$ and $K = 100$ on the data from Figure 2.13. With $K = 1$, the decision boundary is overly flexible, while with $K = 100$ it is not sufficiently flexible. The Bayes decision boundary is shown as a purple dashed line.*

Vecinos más cercanos

Vecinos más cercanos es un método no paramétrico. No asume distribuciones subyacentes en los datos (i.e., datos generados por funciones de densidad que pueden parametrizarse con un número finito de parámetros), sólo asume que inputs similares tendrán outputs similares

Maldición de la dimensionalidad: en altas dimensiones es más difícil encontrar ejemplos cercanos (vean el ej. 4 de la Sección 4.8 de ISLP)



Vecinos más cercanos

Consideraciones:

- Comúnmente las medidas de distancia se ven afectadas por las varianzas las variables (que puede alterarse de manera triviales, por ej. cambiando de unidad de medida), por este motivo las variables suelen ser reescaladas/estandarizadas

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \qquad\qquad x' = \frac{x - \bar{x}}{\sigma}$$

- En vecinos más cercanos (con distancia Euclídea), no hay una manera plenamente satisfactoria de trabajar con atributos categóricos, dos opciones comunes son:
  - Dar un puntaje de 0 si entre ambas observaciones el atributo tiene el mismo valor, y 1 si no lo tiene
  - Usar one-hot-encoding

#	Color
0	Red
1	Green
2	Blue
3	Red
4	Blue

#	Red	Green	Blue
0	1	0	0
1	0	1	0
2	0	0	1
3	1	0	0
4	0	0	1

Vecinos más cercanos

El algoritmo puede adaptarse fácilmente a problemas de regresión (algo que ocurre con muchos algoritmos)

En vez de tomar la clase mayoritaria en N_0, se toma el promedio de la variable a predecir

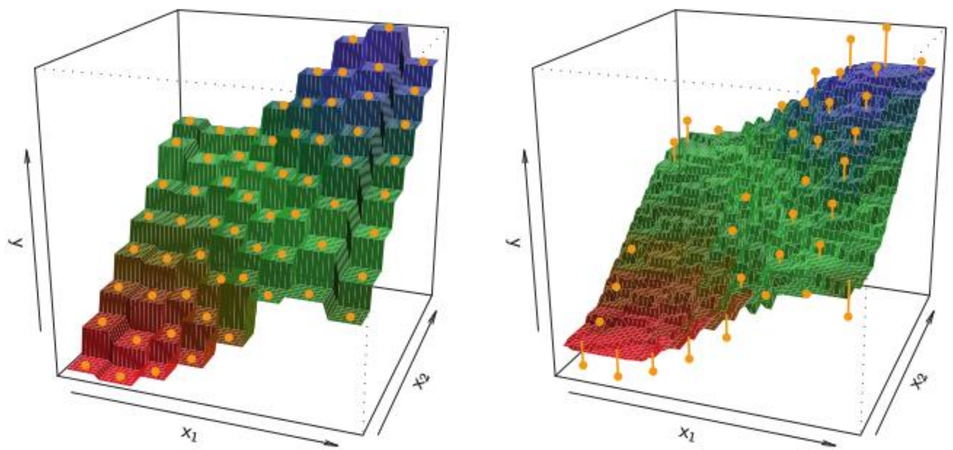


FIGURE 3.16. *Plots of $\hat{f}(X)$ using KNN regression on a two-dimensional data set with 64 observations (orange dots).* Left: *$K = 1$ results in a rough step function fit.* Right: *$K = 9$ produces a much smoother fit.*

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

Vecinos más cercanos

¿Cuándo hace el trabajo más duro Naïve Bayes, al aprender de los datos o al clasificar un nuevo registro? ¿y K-nearest neighbors?

Vecinos más cercanos

¿Cuándo hace el trabajo más duro Naïve Bayes, al aprender de los datos o al clasificar un nuevo registro? ¿y K-nearest neighbors?

Ambos algoritmos pertenecen a dos familias distintas.

- Lazy: no computan el modelo cuando se les dan los datos de entrenamiento y posponen el grueso del cómputo a cuando se los usa para predecir

- Eager: computan el modelo al recibir los datos de entrenamiento, y guardan los patrones aprendidos de los mismos para luego usarlos para predecir

¿Qué guarda Naïve Bayes para clasificar instancias futuras? ¿y K-nearest neighbors?

¿Qué desventaja ven a los algoritmos lazy?

Vecinos más cercanos

Puntos a favor:

- Simple de implementar
- Permite incorporar medidas de distancia que tengan sentido en el dominio que se está trabajando
- Maneja de una manera natural problemas multi-clase
- Puede llegar a tener un buen desempeño de disponer de suficientes datos representativos

Puntos en contra:

- El problema de buscar los vecinos puede ser costoso computacionalmente
- Almacenar el modelo puede resultar muy costoso

Bibliografía

Naïve Bayes

- Bramer, "Principles of Data Mining". Capítulo 2.
- Tan, "Introduction to Data Mining". Secciones 5.3.1, 5.3.2 & 5.3.3. No verlo de ISLP

K-nearest neighbors

- Bramer, "Principles of Data Mining". Capítulo 2.
- ISLP. Capítulo 2, Sección 3.5