Vecinos más cercanos

UCO ONLINE

Método *k Nearest Neighbors*

El algoritmo de *k* vecinos más cercanos, comúnmente llamado kNN por sus siglas en inglés (*k Nearest Neighbors*) es un método que, a diferencia de otros métodos estudiados, no necesita generar un clasificador como tal en un proceso de entrenamiento. En este caso, se almacena el conjunto de datos de entrenamiento, de modo que cuando llega una nueva instancia, se le clasifica tras compararla con las instancias más similares a ella en el conjunto de datos.

En la Figura 1 se muestra un ejemplo de clasificación con kNN. Supongamos un problema con 3 clases. Antes de clasificar las nuevas instancias, habrá que seleccionar un valor de *k*. Este valor indica cuántas instancias del conjunto de entrenamiento se utilizan para clasificar una nueva instancia. En el ejemplo, se utilizan valores de k=1 y k=3 para ilustrar distintos escenarios. Dada una nueva instancia (indicada como una cruz), si se utilizase k=1, se clasificaría como de la clase 1, ya que el patrón más cercano es de esa clase. Sin embargo, si se utilizase un valor de k=3, se clasificaría como de la clase 2, ya que 2/3 de los patrones más cercanos son de la clase 2.

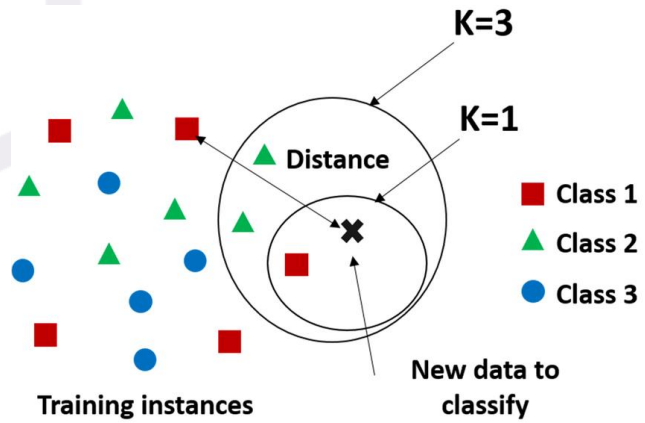

Figura 1. Ejemplo de clasificación de una nueva instancia por kNN para distintos valores de k[1].

Como se observa, kNN es un método muy simple de clasificación, pero aun así es muy potente. Supongamos como ejemplo los datos que aparecen en la Figura 2. Como podemos ver, ambos casos no son linealmente separables, es decir, no podemos obtener un clasificador simple que trazando una línea sea capaz de separar ambas clases correctamente. Sin embargo, un método basado en vecindad, como kNN, sí que sería capaz de clasificar prácticamente cualquier patrón de dicha base de datos correctamente, ya que para clasificar un nuevo dato utiliza únicamente los patrones más cercanos a ese dado.

[1] Fuente: Zmitri, M., Fourati, H., & Vuillerme, N. (2019). Human activities and postures recognition: From inertial measurements to quaternion-based approaches. *Sensors*, *19*(19), 4058.

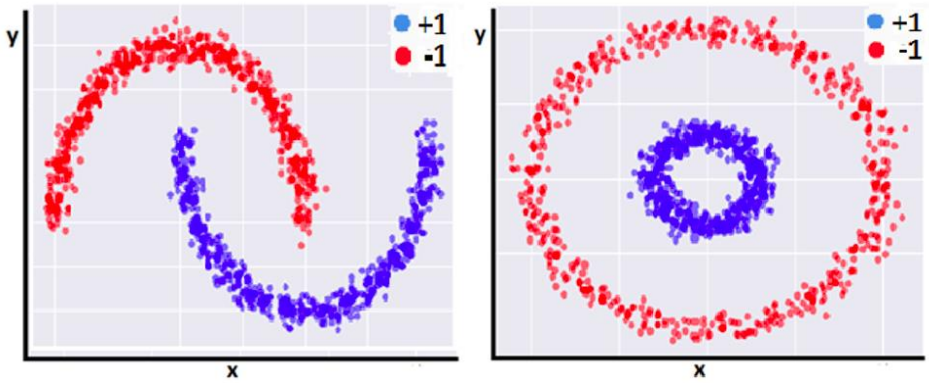

Figura 2. Datos siguiendo distribuciones no linealmente separables[2].

Sin embargo, también existen varias cuestiones a tratar cuando utilizamos kNN, como:

- ¿Qué valor de k escoger?, es decir, ¿cuántos vecinos debería considerar?
- ¿Qué métrica de distancia utilizar?
- ¿Cómo combinamos la información de varios vecinos? ¿Deberían tener todos el mismo peso o influencia en la decisión final?

Función de distancia

Como hemos visto anteriormente, kNN basa la clasificación de un nuevo patrón en el patrón (o patrones) más similares existentes en la base de datos. Sin embargo, ¿cómo medimos o cuantificamos esa similaridad? Para ello utilizamos una métrica de distancia para medirla, conociendo siempre que una función de distancia d devuelve un valor real, y que para unas coordinadas x, y, y z:

- $d(x,y) \geq 0$; y $d(x,y) = 0$ solo si $x = y$
- $d(x,y) = d(y,x)$
- $d(x,z) \leq d(x,y) + d(y,z)$

La distancia utilizada de forma más común es la distancia Euclídea, que muestra la forma usual en que medimos la distancia en la vida real. Se define la distancia Euclídea en la siguiente ecuación, donde $\mathbf{x} = x_1, \ldots, x_m$ e $\mathbf{y} = y_1, \ldots, y_m$ representan los valores para los m atributos de ambos patrones.

$$d_{Euc}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (x_i - y_i)^2}$$

[2] Fuente: https://medium.com/@sachinkun21/using-a-linear-model-to-deal-with-nonlinear-dataset-c6ed0f7f3f51

Antes de calcular la distancia entre patrones, debemos saber que ciertos atributos pueden tener valores mucho mayores que otros, y que los primeros podrían tener una influencia mucho mayor que los segundos en el cálculo de la distancia. Por ejemplo, supongamos dos atributos: ingresos mensuales y edad, y tres patrones:

A. Ingresos = 1.300€ ; Edad = 42 años.
B. Ingresos = 980€ ; Edad = 40 años.
C. Ingresos = 1.200€ ; Edad =26 años.

Si calculamos la distancia euclídea entre algunos de los patrones, podríamos obtener:

- $d_{A,B} = \sqrt{(1300 - 980)^2 + (42 - 40)^2} = 320.01$
- $d_{A,C} = \sqrt{(1300 - 1200)^2 + (42 - 26)^2} = 101.27$

Como vemos, la distancia entre los patrones A y C es mucho menor que la distancia entre A y B. Pese a que A y B tienen prácticamente la misma edad, la dimensión del atributo ingresos es mucho mayor, teniendo una mayor influencia en el cálculo. Para evitar este problema, por lo general será necesario y/o interesante normalizar los datos antes de utilizar un método de clasificación basado en distancias. Para ello, podemos utilizar dos métodos principalmente (ambos igual de válidos) para ajustar nuestros datos originales X en datos normalizados X^*:

- Normalización *min-max*. Normaliza los datos para que los datos de entrenamiento de un atributo X tomen valores en el rango [0, 1].

$$X^* = \frac{X - \min(X)}{\max(X) - \min(X)}$$

- Normalización Z-score. Normaliza los datos como si se tratase de una distribución normal. Por lo general, la mayoría de valores se encontrarán en el rango [-3, 3].

$$X^* = \frac{X - media(X)}{stdv(X)}$$

Supongamos el mismo ejemplo anterior, pero con los datos normalizados siguiendo el método *min-max*. Los patrones y sus distancias quedarían como se muestra a continuación:

A. Ingresos* = 1; Edad* = 1
B. Ingresos* = 0; Edad* = 0.875
C. Ingresos* = 0.6875; Edad* = 0

$$d_{A^*,B^*} = \sqrt{(1 - 0)^2 + (1 - 0.875)^2} = 1.01$$

$$d_{A^*,C^*} = \sqrt{(1 - 0.6875)^2 + (1 - 0)^2} = 1.05$$

Como vemos, tras normalizar los datos y hacer que todos los atributos tengan la misma influencia en el cálculo de la distancia, los valores resultado pueden variar. Ahora, la distancia entre los patrones A y C es ligeramente mayor que la distancia entre A y B.

Por otro lado, la distancia euclídea en principio solo puede aplicarse a atributos de tipo numérico. Si queremos comparar atributos categóricos, podemos definir una función de distancia entre ambos valores categóricos de modo que si ambos son iguales la distancia es 0, y si son distintos su distancia es 1.

Supongamos de nuevo un ejemplo similar al anterior, donde además de los ingresos y la edad, los patrones tienen un atributo de género. Las distancias se calcularían como se indica a continuación:

A. Ingresos* = 1; Edad* = 1; Género = F
B. Ingresos* = 0; Edad* = 0.875; Gen. = F
C. Ingresos* = 0.6875; Edad* = 0; Gen=M

$$d_{A^*,B^*} = \sqrt{(1-0)^2 + (1-0.875)^2 - 0^2} = 1.01$$

$$d_{A^*,C^*} = \sqrt{(1-0.6875)^2 + (1-0)^2 - 1^2} = 1.45$$

Nótese que al realizar la normalización *min-max*, los valores estarán prácticamente todos en el rango [0, 1]; mientras que al realizar la normalización *Z-score*, los valores suelen estar en el rango [-3, 3]. Por tanto, cuando que se utilicen atributos categóricos (y dado que se define la distancia entre estos atributos como 0 o 1), la normalización *min-max* de los atributos numéricos será preferible.

Función de combinación

Para este método, no solo necesitamos definir una métrica de distancia para obtener cuáles son los vecinos más cercanos a un patrón dado, sino también cómo combinar la información de dichos vecinos.

La combinación más básica sería un voto simple sin pesos. En este caso, la clasificación de una nueva instancia vendría dada simplemente por la clase mayoritaria de entre las clases de los k vecinos más cercanos. Es decir, si $k = 3$, y dos de esos 3 patrones son de la clase 2, y otro de la clase 1 (Figura 1), el patrón se clasificará como de la clase 2 (independientemente de la cercanía de cada uno de esos vecinos al patrón). Además, la confianza o probabilidad de pertenencia a dichas clases sería de 0.667 para la clase positiva, y de 0.333 para la negativa.

Sin embargo, el método de voto simple puede producir que se produzcan empates que kNN no sepa cómo resolver. Supongamos el mismo caso de $k = 3$ en un entorno multi-clase, donde los vecinos más cercanos a un patrón dado son de las clases A, B y C respectivamente. En este caso, hay 3 clases candidatas con empate a 1 voto y por tanto existe un empate, y kNN no podría predecir la clase de esa instancia. En estos casos, se podría implementar algún método de desempate para que el método siempre devuelva una clase predicha.

Por otro lado, podemos pensar que aquellos vecinos que están más cerca del patrón dado deberían tener más peso en la decisión final que aquellos más lejanos. Así, se define el método de combinación con pesos, que además permite resolver los empates mucho más fácilmente. En este método, la influencia de una instancia dada es inversamente proporcional al cuadrado de la distancia de dicha instancia a la instancia a clasificar.

Recordemos el ejemplo en la Figura 1, donde pese a que el patrón más cercano era el de la clase 1, se clasificó como de la clase 2 (para k=3). Supongamos que la distancia de la instancia a clasificar al patrón de la clase 1 $d_{X,A} = 0.5$, mientras que las distancias de la instancia a clasificar a ambos patrones de la clase 2 son $d_{\{X,B\}} = 1$ y $d_{\{X,C\}} = 3$; los votos obtenidos para cada clase serían:

- Clase 1: $votos_{X,C_1} = \sum_{i \in C_1} \frac{1}{d_{X,i}^2} = \frac{1}{d_{X,A}^2} = \frac{1}{0.5^2} = 4$
- Clase 2: $votos_{X,C_2} = \sum_{i \in C_2} \frac{1}{d_{X,i}^2} = \frac{1}{d_{X,B}^2} + \frac{1}{d_{X,C}^2} = \frac{1}{1^2} + \frac{1}{3^2} = 1.111$

Como la clase 1 obtiene un mayor peso o número de votos, la instancia en cuestión en este caso se clasificaría como de la clase 1 y no como de la clase 2 (como cuando utilizábamos voto sin peso).

Nótese que en aquellos casos donde la distancia es 0, la inversa estaría indefinida. En estos casos, el método debería considerar la clase mayoritaria de entre las instancias con distancia 0.

En la literatura existen muchas más variantes de kNN y la forma en que se combinan la información dada por cada uno de los patrones cercanos; sin embargo, en esta lección nos centramos en estos dos métodos de combinación más comunes.

Selección de k

Por último, quedaría determinar cómo seleccionar el valor óptimo de k. Sin embargo, no hay una solución que sea la mejor para todos los casos.

Consideremos primero que escogemos un pequeño valor de k. Es posible que la estimación o clasificación esté muy afectada por *outliers* (patrones anómalos) u observaciones ruidosas. Con un valor muy pequeño, por ejemplo k=1, el algoritmo devuelve simplemente la clase del patrón más cercano, lo que podría llevar al algoritmo a sobreentrenar, memorizando los datos de entrenamiento en lugar de ser capaz de generalizar.

Por otro lado, escoger valores de k que no sean pequeños puede tender a suavizar mucho la información del conjunto de datos de entrenamiento. Así, se podría perder información localizada interesante que se pasaría por alto.

Por tanto, la selección del valor de k es un proceso que debe realizar el analista de datos, de forma manual, basándose en la información proveída por los datos. Uno de los procesos más extendidos para permitir que los datos nos ayuden a seleccionar dicho valor de k es seguir un proceso de validación cruzada o *cross-validation*. Así, se podrían probar distintos valores de k sobre datos de entrenamiento y validación, de modo que se minimice el error en los conjuntos de validación. El valor de k que obtenga el menor error podría ser el seleccionado.

Referencias

[Lar14] Larose, D. T., & Larose, C. D. (2014). *Discovering knowledge in data: an introduction to data mining* (Vol. 4). John Wiley & Sons.