K-vecinos más cercanos (KNN)

Ayar Yuman Paco Sanizo
Rafael Villca Poggian

1. Introducción

1.1. Modelos paramétricos vs no paramétricos

Una forma de clasificar modelos es dividirlos entre aquellos que tienen un número fijo de parámetros y aquellos cuyo número varia según el tamaño del conjunto de datos de entrenamiento. A los primeros se los denomina **modelos paramétricos** y a los segundos **modelos no paramétricos**. La ventaja de los modelos paramétricos es que son más rápidos al momento de usarse desde una perspectiva computacional. Sin embargo, tienen la desventaja de tomar supuestos más fuertes sobre la naturaleza de los datos. Por otra parte, los modelos no paramétricos si bien consideran menos supuestos y resultan más flexibles, tienen la desventaja de exigir mayor capacidad de cómputo.

Una vez resumida la diferencia, procedemos a desarrollar el modelo de los K vecinos más cercanos que pertenece al conjunto de modelos no paramétricos.

1.2. Intuición de las estimaciones por KNN

Sea el conjunto de datos de entrenamiento

$$\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n) \mid \mathbf{x}_i \in \mathbb{R}^m, y_i \in G\}$$

Donde,

- $\mathbf{x}_i$ Es un vector que contendrá una entrada del dataset.
- y_i Es la clase correspondiente a la observación i.
- G es el conjunto de posibles valores que puede tomar la variable dependiente y_i, dependiendo de si se realiza una regresión o clasificación, puede ser contínuo o categórico.

Dado un nuevo punto $\tilde{\mathbf{x}}$ del que desconocemos su respuesta y asociada, nos interesa obtener una estimación $\hat{y}$ de la misma. Para este fin, podríamos usar la información de K puntos más cercanos a $\tilde{\mathbf{x}}$ y, bajo un supuesto de similitud con los mismos, obtener una estimación de su respectiva respuesta $\hat{y}$.

Básicamente esta es la forma en que los métodos de K vecinos más cercanos funcionan, métodos de aprendizaje que no hacen uso de parámetros pero que necesitan almacenar todos los datos del conjunto de entrenamiento. Por este motivo, estos métodos también se conocen como métodos basados en memoria. Finalmente, una gran ventaja de los mismos es que podemos usarlos ya sea para problemas de regresión o clasificación.

2. Regresión KNN

2.1. Definición del modelo

Dada una Norma $\|\bullet\|$ definida en $\mathbb{R}^m$ y un punto arbitrario $\tilde{\mathbf{x}} \in \mathbb{R}^m$, si reordenamos el resultado de aplicar la norma a la diferencia de cada punto del dataset con el nuevo punto $\tilde{\mathbf{x}}$, de menor a mayor, quedándonos con las k distancias más pequeñas, obtenemos,

$$N_K(\mathbf{x}) = \|\mathbf{x}_a - \tilde{\mathbf{x}}\| \leq \|\mathbf{x}_b - \tilde{\mathbf{x}}\| \leq \cdots \leq \|\mathbf{x}_k - \tilde{\mathbf{x}}\|$$

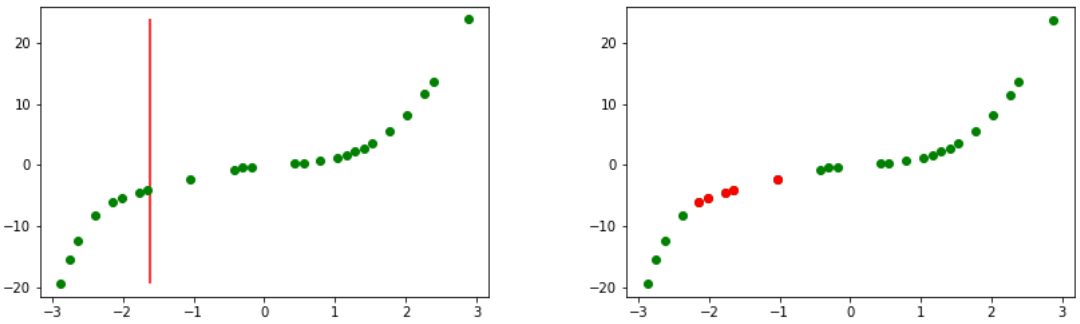

Figura 1 Gráfica de un dataset **izquierda:** el punto $\tilde{\mathbf{x}}$ representado por la línea roja, y **derecha:** los 5 puntos más cercanos en el eje x al punto $\tilde{\mathbf{x}}$.

Si la variable de respuesta y asociada al nuevo punto $\tilde{\mathbf{x}}$ es contínua, podemos obtener su estimación $\hat{y}$ por K-vecinos más cercanos (KNN) usando:

$$\hat{y} = \hat{y}(\mathbf{x}|\mathcal{D}, K) = \frac{1}{K} \sum_{\mathbf{x}_i \in N_K(\mathbf{x})} y_i \tag{1}$$

Donde,

- K: Número de vecinos considerados.
- $N_K(\mathbf{x})$: Conjunto de los K puntos más cercanos a $\mathbf{x}$.

2.2. Algoritmo

El modelo para obtener una estimación por KNN queda incompleto sin una secuencia de pasos lógica para su implementación. Luego, formalizando nuestra intuición, definimos algoritmo de la regresión KNN como sigue.

Algorithm 1 K Vecinos Más Cercanos

```
procedure REGRESION KNN(X, Y, x̃, k)
    n ← longitud de Y
    distancias ← [ . . . ]
    for i = 1 to n do
        distancias[i] ← {distancia(X[i], x̃), Y[i]}
    distancias ← ordenar por distancia mínima distancias
    predicción ← obtener la media de los k más cercanos de Y
    return predicción
```

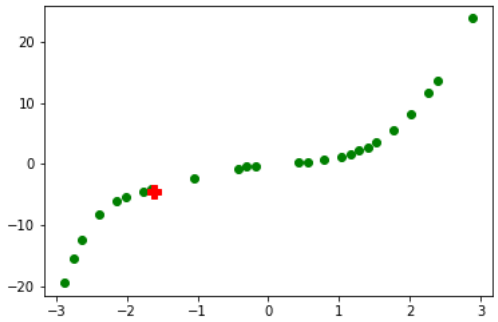

Figura 2 Predicción para el punto $\tilde{\mathbf{x}}$, representada por la cruz roja.

3. Clasificación KNN

3.1. Definición del modelo

De forma similar, si la variable de respuesta es categórica, tal que puede tomar C posibles valores, podemos obtener una estimación $\hat{y}$ como respuesta de $\tilde{\mathbf{x}}$ por K-vecinos más cercanos (KNN) usando:

$$P(y = c_l|\mathbf{x}, \mathcal{D}, K) = \frac{1}{K} \sum_{\mathbf{x}_i \in N_K(\mathbf{x})} I(y_i = c_l) \quad , \quad 1 \leq l \leq C \tag{2}$$

$$\hat{y} = \underset{c \in \{c_1, c_2, ..., C\}}{\operatorname{argmax}} P(y = c|\mathbf{x}, \mathcal{D}, K) \tag{3}$$

Donde,

- K: Número de vecinos considerados.
- $I(.)$: función indicatriz. Es igual a 1 cuando se cumple su condición, 0 en otro caso.
- $N_K(\mathbf{x})$: Conjunto de K vecinos más cercanos a $\mathbf{x}$.

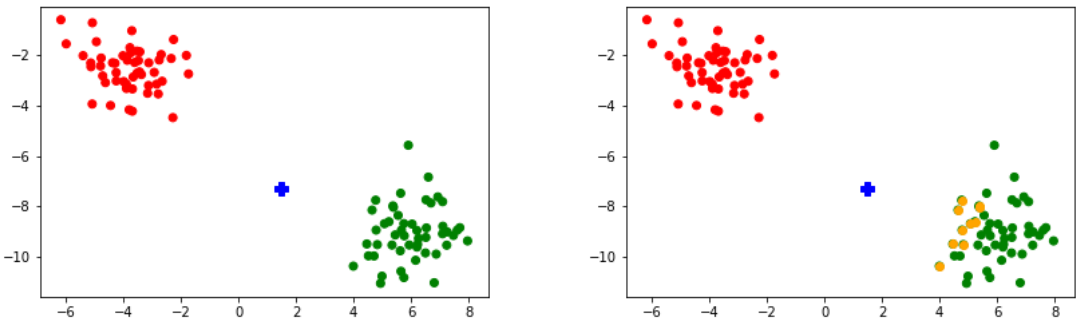

Figura 3 Gráfica de un dataset **izquierda:** el punto $\tilde{\mathbf{x}}$ representado por la cruz azúl, y **derecha:** los 5 puntos más cercanos al punto $\tilde{\mathbf{x}}$ coloreados de naranja.

3.2. Algoritmo

Análogamente, podemos definir el algoritmo de la clasificación por KNN como sigue.

Algorithm 2 Clasificación K Vecinos Más Cercanos

procedure KNN(X, Y, $\tilde{x}$, k)
    $n \leftarrow$ `longitud de` Y
    distancias $\leftarrow [\dots]$
    for i `=1 to` n **do**
        $distancias[i] \leftarrow \{distancia(X[i], \tilde{x}), Y[i]\}$
    distancias $\leftarrow$ `ordenar por distancia mínima` *distancias*
    ocurrencias $\leftarrow$ `contar ocurrencia de clases en los primeros k` *distances*
    predicción $\leftarrow$ `asignar la clase con mayor ocurrencia` *ocurrencias*
    return *predicción*

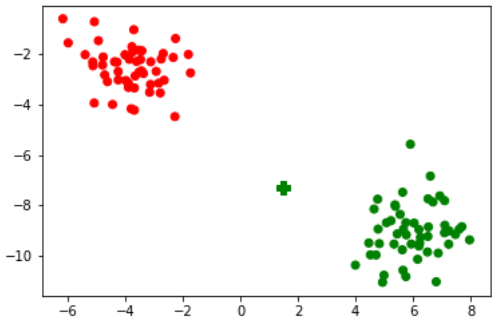

Figura 4 Predicción para el punto $\tilde{\mathbf{x}}$, representada por la cruz, cuyo color verde representa la clase a la que se le asignó el tipo.

4. Medidas de distancia

Notemos que los modelos anteriormente definidos se basan en identificar vecindarios de puntos cercanos. Cuando nuestros datos se representan a partir de atributos numéricos, la idea de cercanía se basa en distancias. De esta forma, podemos evaluar la cercania relativa entre un punto $\mathbf{u}$ y otro $\mathbf{v}$ a partir de medidas de distancia. Esto es, en un conjunto de puntos, podemos decir que $\mathbf{u}$ es más cercano a $\mathbf{v}$, y viceversa, mientras estas distancias sean menores respecto otros puntos. Entre las medidas de distancia más usadas, tenemos las siguientes:

4.1. Distancia Euclideana

Es la métrica más utilizada en aplicaciones donde tengamos que comparar distancias en dimensiones relativamente bajas, consiste en comparar la diferencia en cada una de las dimensiones, elevada al cuadrado.

$$d_{Euclidiana}(\mathbf{u}, \mathbf{v}) = \sqrt{\sum_{j}^{m} (u_j - v_j)^2}$$

4.2. Distancia de Manhattan

Representa cada punto de las distintas dimensiones como cuadras de una ciudad imaginaria, de forma que la métrica representa el número de cuadras en la grilla que se deben recorrer para llegar al destino.

$$d_{Manhattan}(\mathbf{u}, \mathbf{v}) = \sum_{j}^{m} |u_j - v_j|$$

4.3. Distancia de Minkowski

Es una generalización de la distancia euclideana con un valor de $1 \leq p \leq 2$

$$d_{Minkowski}(\mathbf{u}, \mathbf{v}) = \left(\sum_{j}^{m} |u_j - v_j|^p \right)^{\frac{1}{p}}$$

4.4. Distancia de Mahanalobis

Es una generalización de la distancia euclideana que toma en cuenta la varianza de los datos en cada dimensión; a diferencia de la distancia Euclideana, en la cual todos los puntos equidistantes están en una n-esfera alrededor de un punto, en esta métrica los puntos están dispuestos en forma de elipse compensando la diferencia de escalas.

$$d_{Mahanalobis}(\mathbf{u}, \mathbf{v}) = \sqrt{(u - v)^T \Sigma^{-1} (u - v)}$$

4.5. Similaridad del Coseno

Mide el coseno del ángulo entre dos vectores, de forma que si los puntos a comparar están cerca, el coseno entre ambos vectores será más cercano a 1, indicando correlación, caso contrario estará más cerca -1, indicando que están muy separados.

$$d_{Coseno}(\mathbf{u}, \mathbf{v}) = \frac{u \odot v}{\|u\| \, \|v\|}$$

5. Aspectos sobre los K-vecinos más cercanos

- Aunque parezca que los métodos de K-vecinos más cercanos tengan tan solo un parámetro (K), en realidad tienen efectivamente un total de n/K parámetros. Es decir un número de parámetros que incrementa con el tamaño de los datos de entrenamiento y que reduce cuando incrementamos K. Para notar esto consideremos un problema de regresión cuyo conjunto de puntos presenta vecindarios no solapados. De esta forma se tienen n/K vecindarios separables y la predicción en cada uno será el promedio de sus respectivas respuestas.
- Las predicciones se ajustarán mejor a los datos de entrenamientio para valores pequeños de K, presentando de esta forma mayor varianza. Sin embargo, en un extremo esto no es bueno y al contrario puede resultar en un problema traducido en errores altos al generar predicciones con datos de testeo, datos fuera de $\mathcal{D}$. Este problema es denominado sobreajuste (overfitting).
- Podemos obtener estimaciones sesgadas o conservadoras cuando usamos valores grandes de K. Los errores predicción respecto a los datos de entrenamiento incrementarán. En un extremo, también podemos obtener errores de predicción altos al usar datos de testeo. En este caso habremos caído en un problema conocido como subajuste (underfitting).
- Para encontrar un valor óptimo de K, un valor que minimice los errores de testeo, podemos aplicar una técnica denominada validación cruzada.
- Previstos de un conjunto considerable de datos de entrenamiento y una medida de distancia adecuada, los métodos de K-vecinos más cercanos pueden llegar a tener un alto poder de predicción. Sin embargo, a medida que el espacio dimensional de las variables predictoras $\mathbf{x}$ incremente, obtendremos rendimientos cada vez más bajos al usar estos métodos. Este es un problema importante que concierne a varios modelos, tal es su importancia que es considerada una maldición, *la maldición de la dimensionalidad*.

6. La maldición de la dimensionalidad

Consideremos un conjunto de puntos distribuido uniformemente en un hipercubo de tamaño uno. Segmentemos el espacio de forma que tengamos puntos de corte cada $\frac{1}{10}$ de espacio, y que todos los puntos que caigan en un segmento automáticamente se redondean al punto de corte más cercano.

Figura 5: Hiper cubo unidimensional con puntos de corte separados por una distancia de 1/10.

observamos que el punto más lejano de cada una de las marcas está a ese décimo de distancia, entonces, si extendemos el ejemplo a un hipercubo bidimensional (un cuadrado), tenemos que para mantener la misma distancia entre las marcas, debemos debemos agregar muchas más para cubrir el nuevo espacio existente por la nueva dimensión, exactamente tendríamos 100 puntos de corte cada uno separado del otro en ambas dimensiones por una distancia de $\frac{1}{10}$.

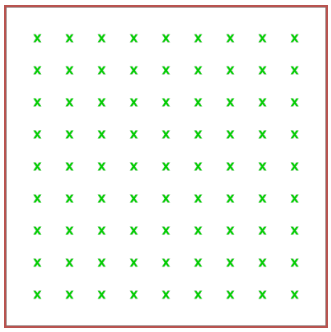

Figura 6: Hiper cubo bidimensional con puntos de corte separados por una distancia de 1/10 en cada dimensión.

si extendemos este razonamiento a más dimensiones, llegamos a la conclusión que la fórmula general para el número de puntos n en m dimensiones sería n^m.

Por lo tanto si queremos que la distancia de la vecindad de puntos se mantenga igual de pequeña a medida que las dimensiones aumentan, la cantidad de datos necesarios crecerá exponencialmente, de esta forma podemos llegar a conclusión que a mayor dimensionalidad, más costoso de ajustar es el modelo.

Otro aspecto de esta maldición es que aquello que parecía cercano en un espacio de pocas dimensiones, resulta más lejano en varias de ellas.

Consideremos un conjunto de puntos distribuido uniformemente en un hipercubo de tamaño uno. Agrupemos un vecindario en un hipercubo pequeño a fin de captar una fracción r de las observaciones. Dado que el hipercubo tiene volumen uno y r es una fracción de la distancia de los puntos como en la $figura5$, la longitud de cualquiera de los lados del hipercubo pequeño en un espacio m-dimensional será igual a $e_m(r) = r^{1/m}$, que se obtiene despejando la longitud de un lado de un cubo de la fórmula para obtener su volumen, en este caso tomando como el "volúmen" a la fracción de la distancia de los puntos r.

Ahora notemos,

- $m = 1\ , \quad r = 0{,}10 : \qquad e_1(0{,}10) = 0{,}10^{1/1} \qquad = \qquad 0{,}10$
- $m = 10, \quad r = 0{,}10 : \qquad e_{10}(0{,}10) = 0{,}10^{1/10} \qquad \approx \qquad 0{,}80$
- $m = 10, \quad r = 0{,}01 : \qquad e_{10}(0{,}01) = 0{,}01^{1/10} \qquad \approx \qquad 0{,}63$

este comportamiento quiere decir que para captar un 10 % de las observaciones, en una dimensión, debemos cubrir el mismo porcentaje del rango la característica correspondiente. Por otra parte, si queremos hacer lo mismo en un espacio de 10 dimensiones, tendremos que cubrir el 80 % del rango de cada variable. Lastimosamente, reducir el fracción r no resulta de mucha ayuda pues, como se puede ver, al reducir la cobertura a 1 % de los datos, el porcentaje de rango a cubrir por variable tan solo reduce a 63 %. Otro aspecto de la maldición de la dimensionalidad es que se requieren muchos más datos de entrenamiento en espacios de varias dimensiones. En concreto, si en una dimensión consideramos que n_1 es una muestra aceptable para realizar estimaciones, en un espacio m-dimensional vamos a requerir una muestra de tamaño n_1^m para representar la misma densidad. Por ejemplo para $n_1 = 100$ y $m = 10$ requeririamos un total de $n_{10} = 100^{10}$ observaciones.

7. Bibliografía

- Hastie, T., Tibshirani, R., & Friedman, J. H. (2008). The Elements of Statistical Learning: Data mining, inference, and prediction.
- Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective.