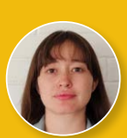
CIENCIAS BÁSICAS
Ciencia Vital, Vol. 4, No. 2, abril-junio 2026
https://doi.org/10.20983/cienciavital.2026.02.bas.02
e0402BAS02

k-NN, la eficiencia de compararte

con tus vecinos

Dr. Víctor Manuel Vázquez Báez*[1]
Estefanía Espinosa Fernández[2]
Mtra. Gabriela Yáñez Pérez[3]
Dra. Dolores García Toral[4]






*1 Benemérita Universidad Autónoma de Puebla https://orcid.org/0000-0002-4980-1832. manuel.vazquez@correo.buap.mx
2 Benemérita Universidad Autónoma de Puebla https://orcid.org/0009-0005-2374-7048
3 Benemérita Universidad Autónoma de Puebla https://orcid.org/0000-0002-4529-5995
4 Benemérita Universidad Autónoma de Puebla https://orcid.org/0000-0001-7944-4242

RESUMEN

Este artículo explica de manera clara y accesible cómo funciona el método k-NN (k-nearest neighbors o "k vecinos cercanos") para imputar, es decir, rellenar datos faltantes en registros obtenidos a lo largo del tiempo. A través de ejemplos y explicaciones sencillas, se describen los conceptos principales involucrados en su implementación computacional, así como los términos técnicos necesarios para comprender su funcionamiento sin requerir conocimientos especializados. El texto también analiza por qué este método ha ganado relevancia frente a técnicas tradicionales, destacando su capacidad para mejorar la precisión en el manejo de información incompleta. Este tipo de herramientas resulta especialmente útil en áreas donde los datos son fundamentales, como la ciencia, la ingeniería, la medicina o el análisis ambiental. Además de presentar los fundamentos del algoritmo, el artículo busca acercar al lector al mundo del aprendizaje automático (Machine Learning), mostrando cómo la inteligencia artificial puede ayudar a resolver problemas reales de forma práctica y eficiente.

Introducción

La inteligencia artificial (IA) y el Machine Learning (ML, aprendizaje automático) son herramientas ampliamente utilizadas en diversas áreas del conocimiento, con aplicaciones que van desde algoritmos para redes sociales hasta análisis de imágenes médicas. Cada vez es más común que las personas tengan contacto con este tipo de recursos, especialmente en ámbitos relacionados con el entretenimiento. Aunque parezca difícil acceder a ellos, lo cierto es que el procedimiento para trabajar con ML es más simple de lo que parece: el sistema se alimenta con un conjunto de entrenamiento (los primeros datos con los que tiene contacto) que utiliza para aprender, para luego generar un modelo a partir de este y evaluarlo con un conjunto de prueba (datos separados previamente para medir el desempeño del modelo).

Una de las razones por las que la inteligencia artificial ha encontrado un espectro tan amplio de aplicaciones es la variedad de algoritmos que engloba. Si bien los principios de funcionamiento son similares, los sistemas de inteligencia artificial están construidos de distintas maneras y constantemente se buscan implementar mejoras a los métodos existentes. Muchas veces las técnicas se inspiran en elementos preexistentes: algunos ejemplos interesantes son las redes neuronales (basadas en las células del cerebro) y los árboles de decisión (inspirados en los árboles reales y sus raíces, ramas y hojas). Incluso existen algoritmos basados en regresión lineal o sistemas de recompensa y castigo. Sin duda, uno de los más intuitivos y fáciles de aplicar es k-NN (k-nearest neighbors o k-vecinos cercanos).

En este artículo se expone información importante sobre dicho algoritmo, que además es uno de los más utilizados, simples y antiguos del Machine Learning. Se exploran conceptos relevantes, su funcionamiento, algunas características importantes, cuándo es recomendable utilizarlo y su origen, pues, aunque parezca que la inteligencia artificial es una herramienta reciente, la historia de k-NN demuestra que su aparición se remonta a varias décadas atrás.

Machine Learning

Para comprender a k-NN, es necesario comenzar hablando de inteligencia artificial y Machine Learning. La inteligencia artificial engloba las técnicas que buscan que las máquinas sean capaces de razonar. El Machine Learning, por otro lado, engloba herramientas que, como su nombre lo indica, tienen el objetivo de que las máquinas puedan aprender algo a partir de un conjunto de datos. Es decir, el Machine Learning podría considerarse un tipo de inteligencia artificial.

El aprendizaje de un sistema de Machine Learning puede ser de diferentes tipos: supervisado (se le proporcionan al sistema las soluciones esperadas) o no supervisado (se espera que el sistema aprenda a partir de los mismos datos). También puede clasificarse como parametrizado (los datos siguen una distribución de probabilidad y generan una función matemática) y no parametrizado (los datos no caen dentro de alguna distribución de probabilidad específica y no

generan una función matemática). Para entender mejor esto último, puede decirse, a grandes rasgos, que una distribución de probabilidad es una representación de las probabilidades de que alguna medición ocurra con base en el valor de dicha medición, tal como la famosa curva de campana que describe una gran cantidad de fenómenos naturales.

Dentro de los algoritmos de aprendizaje no parametrizado, uno de los más populares es k-NN (k-nearest neighbors o k-vecinos cercanos) [1], donde la k es un número arbitrario por determinar. Se trata de un algoritmo que utiliza la información perteneciente a k vecinos cercanos para realizar tareas de clasificación o predicción de valores.

Es importante destacar que trabajar con Machine Learning es similar a generar modelos, es decir, realizar abstracciones que permitan analizar y aprovechar la información contenida en un conjunto de datos para predecir eventos futuros.

Origen

La idea de este algoritmo se atribuye a un trabajo de 1951 de Evelyn Fix y J. T. Hodges, altamente relacionado con el campo de la probabilidad. El propósito de los autores era determinar, a partir de observaciones, si un fenómeno se comporta siguiendo una distribución F o una distribución G, mismas que no se conocen en su totalidad y de las cuales solo se cuenta con algunos valores para ciertos puntos.

Al final del reporte, los autores proponen tomar en cuenta solo un número de puntos cercanos, los suficientes para contener k puntos de una muestra combinada entre valores conocidos en F y en G, para realizar el estimado de un nuevo valor dentro de F y utilizar este mismo conjunto de puntos para la estimación de un nuevo valor en G. Una vez que se completan ambos conjuntos de probabilidades por este procedimiento, se determina cuál de las dos distribuciones marca el comportamiento de las mediciones en cuestión. Dado lo anterior, este procedimiento suele considerarse como una primera versión del algoritmo k-NN [2].

Aunque el estudio nunca fue publicado en medios de fácil acceso al público —solo como un reporte interno en una universidad de la Fuerza Aérea de Estados Unidos y recuperado en 1989 para su publicación en una revista especializada—, el algoritmo cobró popularidad gracias a su simplicidad y precisión. Por eso aún se utiliza y, aunque se ha optimizado y modificado a través de la historia, sigue funcionando bajo los mismos principios.

Generalidades

En la literatura se suele clasificar a k-NN como un algoritmo "perezoso", pues funciona bajo el principio de "con las experiencias viejas, genera datos nuevos". Es decir, un dato de investigación se compara con los datos de entrenamiento a partir de una medida, en este caso la distancia [3]. Esto también implica que todo el cálculo ocurre cuando se realiza una clasificación o predicción; por ello, el método depende en gran medida de la memoria [4].

Al emplear un algoritmo de inteligencia artificial, es importante definir el tipo de problema que se quiere resolver. Existen dos tipos de tareas: clasificación y regresión. Aunque k-NN fue originalmente diseñado para la clasificación, ha demostrado ser eficiente también para resolver problemas de regresión, lo cual lo convierte en una herramienta sumamente atractiva gracias a su versatilidad y practicidad.

Para resolver un problema de clasificación, se alimenta a la computadora con un conjunto de datos correspondientes a diferentes clases o categorías, a fin de que "aprenda" a diferenciar entre elementos de cada clase. Por ejemplo, fotografías de diferentes tipos de autos: deportivos, sedanes, SUV o camionetas. A partir de este aprendizaje, cuando sea alimentada con un dato nuevo, podrá diferenciarlo y asignarlo a la clase que le corresponda. Siguiendo con el ejemplo de los autos, supongamos que se proporciona a la máquina la fotografía de un Tsuru; la computadora procederá a catalogarlo como un sedán.

El problema de clasificar puede complicarse tanto como sea necesario, dependiendo de la cantidad de información que se necesite extraer del conjunto de datos. En nuestro ejemplo, además del tipo, pueden subclasificarse los autos por marca, modelo, uso particular, empresarial o taxi, número de cilindros del motor, entre otros aspectos.

Por otro lado, la regresión consiste en el análisis de un conjunto de datos a partir del cual se busca encontrar una tendencia y una desviación respecto a ella. Una vez identificadas, es posible calcular una curva o función teórica que permita conocer valores en puntos nuevos o fuera del conjunto inicial. En otras palabras, imaginemos que Pepito, al salir de la escuela, camina hacia su casa y llega en 15 minutos si no se distrae, pero tarda 20 minutos si se detiene a platicar con sus amigos. Como la mayoría de las veces camina directo a casa, el trayecto de 15 minutos representa la tendencia, mientras que la desviación es de 5 minutos. Por lo tanto, si tuviéramos que predecir el tiempo que tardará en llegar a casa, podríamos decir que tardará aproximadamente 15 minutos.

El algoritmo se compone de tres pasos primordiales:

Calcular la distancia entre el punto de investigación y los puntos de consulta.

Encontrar los vecinos cercanos.

Realizar la clasificación o regresión consultando a los k-vecinos [5].

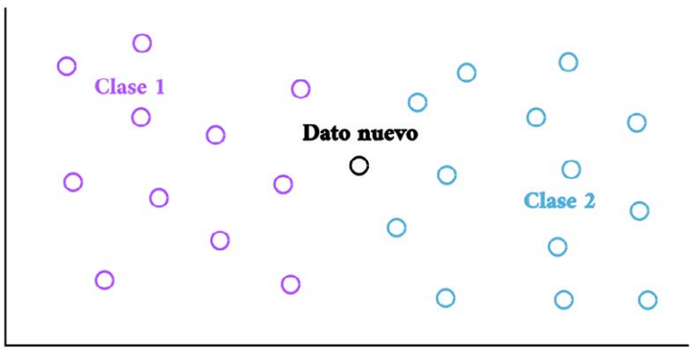


Figura 1. Tarea de clasificación simple.

La precisión de k-NN puede verse seriamente afectada debido a la presencia de ruido en los datos (variaciones aleatorias que dificultan el análisis, como errores de medición o valores extremos), atributos irrelevantes (variables que no aportan información útil al aprendizaje) o un incorrecto escalado de los atributos (transformación de los datos para que tengan un rango o escala similar, especialmente cuando existen unidades distintas) [6]. Por ello, el preprocesamiento debe realizarse cuidadosamente.

Además, es necesario definir adecuadamente los hiperparámetros, es decir, variables internas del algoritmo que toman valores óptimos para maximizar la precisión de la clasificación o regresión calculadas por k-NN.

Primero, es importante conocer el problema para asegurar una correcta elección del algoritmo. Aunque k-NN puede utilizarse tanto para clasificación como para regresión —siendo más común el primer caso—, su funcionamiento cambia ligeramente: para clasificar, consulta a los puntos cercanos y devuelve la etiqueta que más se repite; en regresión, también consulta a los vecinos cercanos, pero realiza una especie de promedio para predecir un valor.

Definiendo el número de vecinos

Probablemente, el parámetro más importante por definir es el valor de k, es decir, con cuántos vecinos cercanos se comparará el punto de interés. Para establecerlo, es importante tomar en cuenta dos conceptos: sobreajuste (los datos se ajustan tanto al conjunto de prueba que no

logran generalizar correctamente) y subajuste (el modelo no realiza buenas predicciones en absoluto).

Si k es un número muy grande, puede generarse sobreajuste; pero si es demasiado pequeño, el algoritmo podría no tomar en cuenta la tendencia general y convertirse en un mal modelo [6]. Asimismo, a mayores valores de k, más grandes son las clases en cuanto a rango, lo que genera un efecto de reducción de ruido, aunque la clasificación puede resultar menos precisa.

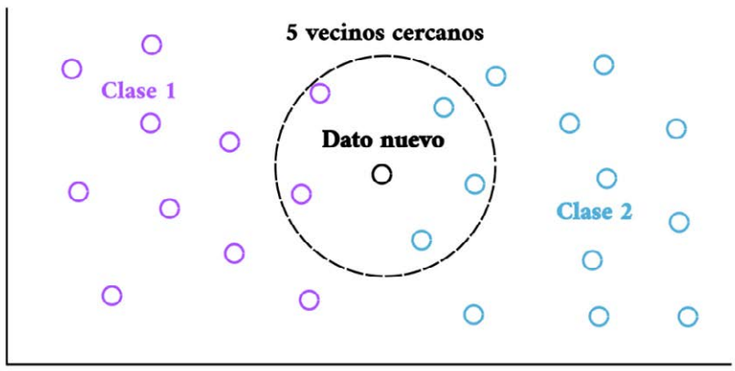


Figura 2. Cinco vecinos más cercanos al punto de consulta.

De manera que k depende del número y tipo de datos. Usualmente se define mediante validación cruzada, que consiste en dividir el conjunto de entrenamiento en dos o más subconjuntos para entrenar y poner a prueba distintos hiperparámetros, con la finalidad de identificar cuál ofrece un mejor desempeño [6]. Adicionalmente, se recomienda utilizar valores impares de k para evitar empates entre clases [7].

También es importante definir la participación de cada vecino, pues el algoritmo puede funcionar de dos maneras: realizar predicciones según el voto de la mayoría o asignar diferentes pesos a los votos dependiendo de la distancia respecto al punto de interés [4].

Distancias

Como el algoritmo consulta puntos cercanos para realizar predicciones, la distancia es otro factor importante por establecer, pues existen diferentes tipos, siendo la más común la euclidiana.

La distancia euclidiana es la más simple de comprender, ya que corresponde directamente a la noción cotidiana de distancia: la línea recta entre el punto de consulta y el punto investigado. También existe la distancia Manhattan, utilizada en situaciones donde es necesario moverse a través de un sistema cuadriculado, de manera similar a como se transita por las calles de una ciudad.

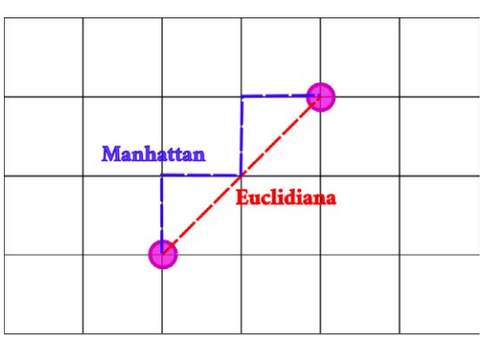


Figura 3. Comparación entre las distancias Manhattan (azul) y euclidiana (rojo).

Otro ejemplo es la distancia Minkowski, en la que se define un parámetro p, lo que la vuelve más flexible. Si p = 1, es equivalente a Manhattan; si p = 2, es equivalente a la euclidiana [8]. Existen otros tipos de distancia, pero estos tres son los más comunes.

Ventajas y desventajas

Como cualquier algoritmo, k-NN funciona mejor bajo condiciones específicas, por lo que es importante conocer sus ventajas y desventajas para asegurarse de que es una opción viable para tratar el conjunto de datos de interés.

Ventajas:

- Fácil de implementar y comprender.
- Tiene pocos hiperparámetros.
- Puede funcionar bien para clasificación multiclase (instancias con más de una etiqueta).
- Se adapta fácilmente a nuevos datos.

Desventajas:

- Sensible al ruido.
- No escala bien con grandes cantidades de datos.
- No genera como salida un modelo matemático para analizar.
- Alto costo computacional.
- Un mal valor de k puede generar problemas.
- [4], [5], [7]

Con un correcto tratamiento, las desventajas pueden ser superadas por los beneficios de k-NN, por lo que cuenta con múltiples aplicaciones con las que una persona puede encontrarse fácilmente y probablemente con frecuencia.

¿Dónde se utiliza?

Suele utilizarse en tareas que requieren alta precisión y no necesitan como salida una fórmula matemática para ser analizada por un humano. Adicionalmente, es recomendable para conjuntos que no tengan demasiados datos. Algunos casos específicos son:

Algoritmos de recomendación para servicios de streaming.

Reconocimiento facial.

Detección de plagio.

Detección de enfermedades según algunos síntomas sutiles.

Clasificación de individuos según su comportamiento y respuesta a estímulos en áreas como la psicología y la sociología [7].

Implementar este algoritmo tampoco es complicado. Una de las maneras más simples requiere utilizar Python y la biblioteca Scikit-learn, en particular los módulos KNeighborsClassifier y KNeighborsTransformer (más información en [9]). Aunque es recomendable consultar a detalle la documentación correspondiente, así como los principios básicos de Machine Learning, para obtener resultados coherentes.

Conclusión

Como puede concluirse a partir de lo expuesto, k-NN se ha convertido en un clásico dentro de las herramientas de inteligencia artificial gracias a su practicidad. No obstante, es importante destacar que emplearlo presenta ventajas y desventajas, por lo que resulta fundamental conocer bien el conjunto de datos y el problema que se quiere resolver. Esto no solo permite saber

si es el algoritmo adecuado, sino también ajustar correctamente sus hiperparámetros y obtener resultados satisfactorios.

Referencias

[1] F. Imam, P. Musilek y M. Z. Reformat, "Parametric and Nonparametric Machine Learning Techniques for Increasing Power System Reliability: A Review", Information, vol. 15, no. 1, p. 37, ene. 2024. doi: https://doi.org/10.3390/info15010037

[2] B. W. Silverman y M. C. Jones, "E. Fix and J. L. Hodges (1951): An Important Contribution to Nonparametric Discriminant Analysis and Density Estimation: Commentary on Fix and Hodges (1951)", International Statistical Review, vol. 57, no. 3, p. 233, dic. 1989. doi: https://doi.org/10.2307/1403796

[3] H. Díaz-Barrios, Y. Alemán-Rivas, L. Cabrera-Hernández, A. Morales-Hernández, M. Chávez-Cárdenas y G. Casas-Cardoso, "Algoritmos de aprendizaje automático para clasificación de Splice Sites en secuencias genómicas", SciELO, 2015. [En línea]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2227-18992015000400012

[4] IBM, "What is the k-nearest neighbors algorithm?", IBM, 2022. [En línea]. Disponible en: https://www.ibm.com/topics/knn

[5] J. Sun, W. Du y N. Shi, "A Survey of kNN Algorithm", Information Engineering and Applied Computing, 2018. [En línea]. Disponible en: https://www.researchgate.net/publication/348305327_A_Survey_of_kNN_Algorithm

[6] R. Nisbet, J. Elder y G. Miner, "Classification", en Handbook of Statistical Analysis and Data Mining Applications. Elsevier, 2009, pp. 235–258. doi: https://doi.org/10.1016/B978-0-12-374765-5.00011-5

[7] United States Artificial Intelligence Institute, "Understanding KNN Algorithm and Its Role in Machine Learning", 2023. [En línea]. Disponible en: https://www.usaii.org/ai-insights/understanding-knn-algorithm-and-its-role-in-machine-learning

[8] T. Davi, "Understanding Different Distance Measures", LinkedIn, 2023. [En línea]. Disponible en: https://www.linkedin.com/pulse/understanding-different-distance-measures-tiago-davi-1f

[9] Scikit-learn, "API Reference - sklearn.neighbors: Nearest Neighbors". [En línea]. Disponible en: https://scikit-learn.org/stable/modules/classes.html#module-sklearn.neighbors

[10] F. Pedregosa et al., "Scikit-learn: Machine learning in Python", Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011. [En línea]. Disponible en: https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf