RESUMEN DE CLASIFICADORES RÁPIDOS BASADOS EN EL ALGORITMO DEL VECINO MÁS CERCANO

SUMMARY OF FAST CLASSIFIERS BASED ON THE NEAREST NEIGHBOR ALGORITHM

Selene Hernández Rodríguez* y María Patricia Torrijos Muñoz**

RESUMEN

Actualmente, en diferentes ciencias como la medicina, las geociencias, la astronomía, entre otras, la tarea de clasificación supervisada ha dado solución a muchos problemas importantes. Uno de los algoritmos de clasificación supervisada más utilizados ha sido k vecinos más cercanos (o k Neares Neighbors, k-NN), el cual ha mostrado ser un algoritmo simple, pero efectivo. El algoritmo k vecinos más cercanos realiza una comparación exhaustiva entre el nuevo objeto a clasificar y todos los elementos del conjunto de entrenamiento. Sin embargo, cuando el conjunto de entrenamiento es grande, este proceso es costoso y en

ABSTRACT

Nowadays, in different fields such as Medicine, Geosciences, Astronomy, among others, the supervised classification task has been a solution of many important problems. One of the most applied supervised classification algorithms has been k nearest neighbors (k-NN), which is a simple yet effective algorithm. The k-NN algorithm performs an exhaustive comparison between the query and every prototype in the training set, in order to find the k nearest neighbors of the query. However, when the training set is large or the comparison function is expensive, the exhaustive search becomes impractical or inapplicable in some cases. For this

*Maestra y docente adscrita al Departamento de Ciencias Básicas del Instituto Tecnológico de Puebla, México; pertenece al área de Matemáticas dentro de este instituto. selene.hernandez@puebla.tecnm.mx

**María Patricia Torrijos Muñoz. Maestra y docente adscrita al Departamento de Ciencias Básicas del Instituto Tecnológico de Puebla, México; pertenece también al área de Tutorías y a un Cuerpo Académico dentro del instituto. Se especializa en el área de Probabilidad y Estadística. mariapatricia.torrijos@itpuebla.edu.mx

algunos casos esta búsqueda exhaustiva se vuelve un proceso muy lento o inaplicable. Para agilizar el proceso de clasificación y omitir comparaciones, se han propuesto en los últimos años clasificadores rápidos basados en el algoritmo del vecino más cercano (Fast k-NN). La mayoría de estos algoritmos Fast k-NN se basan en las propiedades métricas de la función de distancia para omitir comparaciones o bien otras heurísticas.

PALABRAS CLAVE: regla del vecino más cercano, clasificadores rápidos basados en el algoritmo del vecino más cercano

reason, in the last years many approaches have been proposed to accelerate the search process performed by k-NN (called fast k-NN). Most of fast k-NN approaches rely on metrical properties of the distance function to avoid prototype comparisons, or other heuristics.

KEYWORDS: Nearest neighbor rule, Fast k-NN / k-MSN classifiers

DOI: https://doi.org/10.5281/zenodo.6963998

RECIBIDO: 8 de junio, 2022
ACEPTADO: 30 de julio, 2022

1. INTRODUCCIÓN

El algoritmo de clasificación supervisada, basado en la búsqueda del vecino más cercano (NN) (Cover & Hart, 1967), ha sido ampliamente utilizado debido a su simplicidad y capacidad para resolver problemas complejos de reconocimiento de patrones. Este algoritmo está basado en un conjunto de objetos de entrenamiento T, previamente clasificado.

$$T = \{Z_i \mid 1 \le i \le N\} \quad (1)$$

Donde N es el número de objetos. El objeto Zi está descrito por d atributos en un espacio $\mathcal{R}^d$.

$$Z_i = [x_1(Z_i), x_2(Z_i), \ldots, x_d(Z_i)] \quad (2)$$

Dado un nuevo objeto Q a clasificar, el objetivo del clasificador es encontrar el objeto NN en el conjunto T que minimice la función de distancia Dist con el objeto Q.

$$NN = \min_{Z_i \in T} Dist(Z_i, Q) \quad (3)$$

Para encontrar O_{NN} se requiere una comparación exhaustiva entre Q y cada objeto del conjunto T. Una vez encontrado el vecino más cercano, se asigna la clase de este objeto a Q. Una generalización de este método es el algoritmo de k vecinos más cercanos (k-NN). En este caso, la búsqueda se realiza de la misma forma que en NN, salvo que en lugar de encontrar un solo objeto, se buscan los k más cercanos. Finalmente, la clase del objeto Q se elige por votación de los k objetos más cercanos.

Los clasificadores rápidos k-NN han sido útiles para resolver diferentes problemas como compresión de datos (Gersho & Gray, 1991), bioinformática (Niijima & Kuhara, 2005), recuperación de información (Deerwester et al., 1990) y reconocimiento de caracteres (Stelios & Bakamidis, 1993), entre otros. En todas estas interesantes aplicaciones, el algoritmo del vecino más cercano (k-NN) ha dado buenos resultados. Sin embargo, con el desarrollo de la tecnología y la capacidad de los dispositivos de almacenamiento, cada vez se requiere procesar cantidades de información más grandes. Tal es el caso de aplicaciones médicas para el diagnóstico de enfermedades basado en una gran cantidad de imágenes.

Otra aplicación interesante y muy utilizada actualmente es la clasificación de objetos web como imágenes, libros, documentos, música, entre otros elementos web utilizados en páginas sociales o de comercio electrónico. En este caso, los sistemas recomendadores utilizan algoritmos de filtrado colaborativo, los cuales hacen uso del algoritmo del k vecinos más cercanos para filtrar preferencias de usuarios similares.

Estos algoritmos utilizan información de usuarios similares o elementos similares para recomendar nuevos elementos a una persona, lo cual es una tarea compleja, ya que estos sitios tienen una gran cantidad de usuarios y elementos (o ítems) (Lucas, et al., 2013). Por esta razón, en los últimos años se han propuesto muchos enfoques para acelerar el proceso de búsqueda realizado por k-NN, que se denominan

clasificadores rápidos basados en el algoritmo del vecino más cercano (Fast k-NN).

La mayoría de los clasificadores rápidos Fast k-NN se basan en heurísticas, que involucran propiedades métricas de

Figura 1. Clasificación general de los clasificadores rápidos basados en el algoritmo del vecino más cercano (Fast k-NN)

la función de comparación y/o algunas estructuras como árboles o matrices de distancia entre elementos, las cuales son creadas en una fase de pre-procesamiento. En la Figura 1 se muestra una clasificación general de algunos métodos de clasificación rápida basados en el algoritmo del vecino más cercano.

Los clasificadores rápidos k-NN pueden ser exactos o aproximados. El objetivo de los clasificadores k-NN rápidos exactos es reducir el número de comparaciones garantizando que se devolverán exactamente los mismos k vecinos más cercanos que se encontrarían utilizando el exhaustivo k-NN y, por lo tanto, obteniendo la misma calidad de clasificación. El objetivo de los clasificadores k-NN rápidos aproximados es reducir el número de comparaciones, y reportar los k vecinos más cercanos encontrados hasta el momento, tratando de mantener la calidad de clasificación original obtenida por el exhaustivo k-NN.

Existen tres aspectos a considerar de los clasificadores rápidos basados en el algoritmo del vecino más cercano (Fast k-NN), los cuales son:

1. *Tiempo de pre-procesamiento.* La mayoría de los enfoques están basados en un procesamiento previo del conjunto de T para generar una estructura de datos que permita lograr una búsqueda más rápida del vecino más cercano. Sin embargo, se espera que dicho procesamiento no sea muy complejo, ya que sería inaplicable en problemas con muchos objetos.
2. *Espacio de almacenamiento.* Se espera también que la estructura generada en el paso de pre-procesamiento se pueda almacenar en un espacio lineal respecto al tamaño del conjunto T.
3. *Tiempo de búsqueda.* Dado un nuevo objeto a clasificar, en los métodos enfocados a la búsqueda rápida se espera que se reduzca el tiempo de búsqueda con respecto al de la búsqueda exhaustiva.

2. CLASIFICACIÓN DE MÉTODOS FAST K-NN APLICABLES A DATOS NUMÉRICOS

2.1. Fast k-NN basados en distancias parciales

El enfoque basado en distancias parciales se ha propuesto para reducir la complejidad del algoritmo k-NN. Este enfoque (Cheng et al., 1984) se basa en la idea de que la distancia parcial es la distancia entre dos prototipos, pero utilizando parte de sus atributos o dimensionalidad. Así, durante el cálculo de la distancia, entre el prototipo a clasificar y un prototipo en el conjunto de entrenamiento, cada vez que se añade un atributo a calcular, se compara la suma parcial obtenida con la distancia al vecino más cercano ($current_{NN}$) encontrado hasta el momento. Si esta distancia ya excede la mínima encontrada hasta el momento (entre el prototipo a clasificar o consultar y el vecino más cercano encontrado hasta el momento, $current_{NN}$), entonces no se terminan de comparar los dos prototipos utilizando todos sus atributos, dado que al considerar más atributos, el valor

de la distancia solo puede aumentar o permanecer igual. Por lo que se puede descartar este elemento como posible vecino más cercano.

En (Hwang & Wen, 2002) se aplicó una técnica de reducción de dimensionalidad al conjunto de entrenamiento (utilizando *wavelets*), antes de aplicar la técnica de distancias parciales. En (Athitsos et al., 2005) se introduce un método aproximado basado en distancias parciales. En este trabajo se introduce un valor de confianza (p). De acuerdo con este método, si los prototipos p más cercanos, utilizando parte de la dimensionalidad de los atributos pertenecen a la misma clase, entonces al prototipo de consulta (Q) se asigna a la misma clase. Si no se cumple esta condición, se aumenta la dimensionalidad de los prototipos y se repite el proceso de comparación.

En la búsqueda exhaustiva, el cálculo de la distancia se realiza atributo por atributo, que es de orden O(Nd), donde N es el número de prototipos y d es el número de atributos que describen los prototipos. Se puede notar que este orden es proporcional al tamaño de N y d, por esta razón, el tiempo requerido aumenta con la dimensionalidad de los atributos. En el enfoque de distancias parciales, se espera que el orden de búsqueda sea O (Nd'), donde d' << d.

2.2. Fast k-NN *classifiers* basados en pre-ordenamiento del conjunto de entrenamiento

En el enfoque desarrollado en (Fisher & Patrick , 1970) se presenta una fase de preprocesamiento, en la que el conjunto de entrenamiento se ordena de tal manera que un prototipo tiende a estar lejos de su predecesor en una lista ordenada. Mediante esta lista ordenada y una matriz que almacena las distancias entre prototipos del conjunto de entrenamiento, se pueden omitir algunas comparaciones de prototipos, basadas en la propiedad de desigualdad triangular. Sin embargo, el proceso de ordenar el conjunto de entrenamiento es complejo, ya que requiere comparar cada elemento con el resto de elementos del conjunto de entrenamiento. Cuando el conjunto de entrenamiento es grande, este proceso se vuelve inaplicable. Por esta razón en Wolfson (1990) se presenta una modificación de Fisher y Patrick (1970).

En Bandyopadhyay y Maulik (2002) se propone reordenar los prototipos en el conjunto de entrenamiento, de acuerdo con sus distancias contra un punto fijo, que es un punto no existente en T, creado con los valores mínimos de cada atributo en los prototipos del conjunto de entrenamiento simulando un punto de origen. En el trabajo de estos mismos autores también se propone almacenar al vecino más cercano de cada prototipo en el conjunto de entrenamiento. En la fase de clasificación, la distancia entre los prototipos en el conjunto de entrenamiento, el punto de referencia de origen y la matriz que almacena vecinos más cercanos de cada prototipo en T se utilizan para ubicar los prototipos cercanos al prototipo a clasificar (Q) en un hipercubo, con el fin de reducir el número de comparaciones de prototipos.

En Li y Malik (2016) se presenta un clasificador rápido k-NN, llamado *Dynamic Continuous Indexing* (DCI). El algoritmo DCI se centra en el problema de la dimensionalidad intrínseca. En este trabajo, los autores muestran que su algoritmo no se ve afectado exponencialmente cuando crece la dimensionalidad de los prototipos. En la fase de preprocesamiento de este clasificador se construyen una serie de índices, cada uno de los cuales ordena los prototipos del conjunto de entrenamiento a lo largo de una cierta dirección aleatoria e impone un orden de todos los prototipos. Cada índice se construye de modo que dos prototipos con rangos similares en el orden asociado (lista) están cerca a lo largo de cierta dirección. En Ke y Jitendra (2017) se presenta una mejora de Li y Malik (2016) denominada ICD priorizada. En esta mejora se asigna una prioridad a cada lista de índices que se utiliza para determinar qué índice procesar en la próxima iteración.

En K. He et al. (2013) se presenta un clasificador k-NN rápido, llamado k-Means Hashing (KMH), que crea códigos binarios a partir del conjunto de entrenamiento extraídos utilizando el algoritmo k-Means y parte de la dimensionalidad de los atributos. En este algoritmo se realiza cierto proceso de optimización de código binario, que se ve afectado cuando aumenta la dimensionalidad de los prototipos. Por esta razón en Yalin et al. (2018) se propone un método de *hashing* mejorado apilado, k-Means Hashing

(SKMH), que utiliza la búsqueda de menor dimensión.

2.3. Fast k-NN basados en la proyección de prototipos

Los clasificadores k-NN rápidos basados en la proyección de prototipos utilizan el espacio geométrico de la representación de prototipos y la propiedad de desigualdad triangular para reducir comparaciones entre prototipos. Así, cada atributo de los prototipos se representa en un eje. En (Friedman, et al., 1977) se propone seleccionar los prototipos más cercanos a Q utilizando un umbral (±u) en cada eje. Finalmente, se selecciona el eje con menos prototipos dentro del umbral para realizar una búsqueda exhaustiva entre Q y los prototipos seleccionados (más cercano a Q en este eje).

En Yunck (1976) este método se extiende utilizando d listas ordenadas del eje de proyección de atributos, por lo que se localizan los prototipos más cercanos a Q en un hipercubo. Sin embargo, un inconveniente de este método es que el costo computacional requerido para encontrar los prototipos en el hipercubo es considerablemente alto. Por esta razón, en Nene y Nayar (1997) se presenta una mejora de este método que consiste en la creación de una estructura de datos que reduce el número de operaciones requeridas para localizar los prototipos en el hipercubo.

En Orchard y Michael (1991) se presenta un clasificador k-NN rápido para prototipos numéricos. El clasificador propuesto por Orchards, requiere calcular y almacenar las distancias entre los prototipos del conjunto de entrenamiento en una fase de preprocesamiento. En la

fase de clasificación en este algoritmo se selecciona aleatoriamente un prototipo (p), luego se calcula la distancia (dist) al prototipo de consulta (Q).

2.4. Fast k-NN basados en estructuras de árboles

La mayoría de los clasificadores rápidos k-NN basados en particionamiento

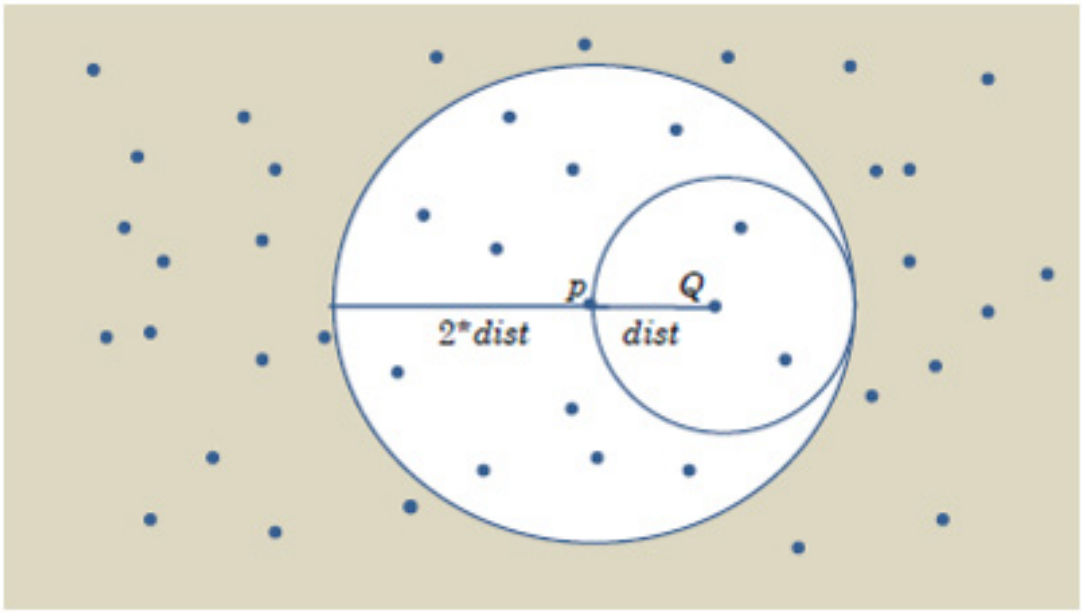

Figura 2. Interpretación gráfica del criterio de eliminación utilizado en Orchard y Michael (1991)

Con base en la propiedad de desigualdad triangular, los prototipos que están a una distancia mayor de 2 × dist del prototipo p se eliminan como posibles vecinos más cercanos. Esta información es conocida, ya que en la matriz de distancias creada en la fase de preprocesamiento, se almacenan las distancias entre p y el resto de elementos del conjunto de entrenamiento. Después de descartar los elementos que están fuera del círculo con diámetro 2 × dist y centro en p, se selecciona un nuevo prototipo y se compara con Q. El vecino más cercano de Q se actualiza y se utiliza para descartar elementos de nuevo (ver Figura 2).

construyen una estructura de árbol en una etapa de preprocesamiento, la cual se crea particionando/dividiendo recursivamente el conjunto de entrenamiento. En la etapa de clasificación se utilizan algoritmos de recorrido del árbol para encontrar los k vecinos más cercanos. Además, se utilizan reglas de poda basadas en la desigualdad triangular para descartar nodos durante el recorrido del árbol y de esta manera realizar menos comparaciones que el clasificador k-NN exhaustivo.

Uno de los primeros clasificadores rápidos k-NN que utilizó una estructura de árbol fue propuesto en 1975 por Fukunaga y Narendra (FN); en los últimos años

han surgido diferentes modificaciones del clasificador FN. El clasificador FN genera una estructura de árbol particionando recursivamente el conjunto de entrenamiento. Para generar las particiones/clústeres se utiliza el algoritmo K-Means. En [16] se propone crear un árbol con tres niveles de profundidad, en el cual cada nodo p del árbol tiene cuatro características, las cuales son: el conjunto de prototipos correspondientes a p (S_p), el número de prototipos en p (N_p), el centro de p (M_p) y finalmente la distancia máxima entre el centro y los prototipos de p (R_p). Para clasificar un nuevo prototipo Q, FN recorre el árbol. Durante el recorrido, se utilizan dos reglas para descartar nodos o prototipos, las cuales están basadas en la desigualdad triangular. La primera regla se utiliza para descartar nodos en los cuales no puede estar el vecino más cercano. Para cada nodo descendiente de p (nodo actual), la primera regla de poda es la siguiente:

$$dist(Q, current_{NN}) + R_p < dist\,(Q, M_p) \quad (1)$$

Donde $current_{NN}$ es el vecino más cercano a Q encontrado hasta ahora y dist es la función de distancia. En este caso, los nodos que satisfacen la condición (1) no pueden contener un prototipo más cercano a Q que el $current_{NN}$ (como el nodo p representado en la Figura 3) y, por lo tanto, esos nodos se eliminan.

La segunda regla de poda se aplica a los prototipos que pertenecen a un nodo hoja del árbol, con el fin de decidir si calcular o no la distancia de Q a los prototipos

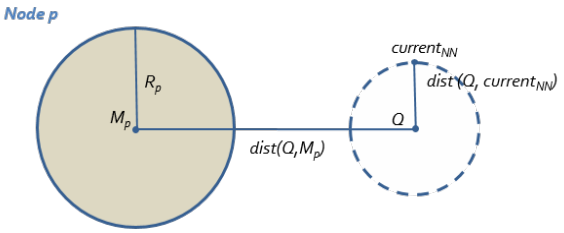

Figura 3. Interpretación gráfica del criterio de poda utilizado en el clasificador FN

del nodo hoja. La regla de poda para cada prototipo $O_i \in S_p$ es:

$$dist(Q, current_{NN}) + dist\,(O_i, M_p) < dist\,(Q, M_p) \quad (2)$$

Los prototipos que satisfacen la condición (2) no pueden estar más cerca de Q que el vecino más cercano actual y, por lo tanto, no se calcula su distancia contra Q. El proceso termina cuando todos los nodos del árbol han sido recorridos o eliminados y, en tal caso, el resultado es el vecino más cercano actual. En los últimos años se han propuesto algunas modificaciones del clasificador FN teniendo en cuenta dos aspectos, el primero de los cuales es la construcción de un árbol que lleve a un proceso de clasificación más rápido. El segundo aspecto ha sido diseñar nuevos algoritmos de búsqueda del vecino más cercano en la etapa de clasificación. En Kalantari y McDonald (1983) se presenta una modificación del clasificador FN. La diferencia es que se modifica el algoritmo para generar la estructura de árbol. En este caso, se construye un árbol binario y cada hoja representa un solo prototipo del conjunto T. Para clasificar un nuevo elemento, se utiliza el mismo algoritmo del clasificador FN.

En Gómez-Ballester et al. (2006) se propone un clasificador (GB), el cual es una variante de FN, en la que se evalúan algunas modificaciones en la regla de poda. En este caso, se comparan tres reglas de poda para la eliminación de nodos durante el recorrido del árbol, la primera de las cuales es la regla de poda FN, la segunda regla está basada en la información de nodos hermanos (Sibling-based pruning rule, SBR), para lo cual se verifica la siguiente desigualdad:
Donde e es el prototipo (contenido en el nodo hermano de p) más cercano al centro (M_p) del nodo p. De acuerdo a esta regla, es posible descartar nodos hermanos sin comparar su centro contra Q. Sin embargo, se requiere calcular y almacenar

$$\text{dist}(e, M_p) \geq \text{dist}(Q, M_p) + \text{dist}(Q, \text{current}_{NN}) \quad (3)$$

en una etapa de preprocesamiento la distancia entre el centro de cada nodo y el prototipo más cercano ($current_{NN}$) de su nodo hermano.

En la Figura 4 se muestra un ejemplo, en el cual el nodo p2 puede ser descartado, sin haber comparado Q contra el centro de p2, como hubiera sido hecho por el clasificador FN. Esta regla es llamada Generalized Pruning Rule (GR).

En Oncina et al. (2007) se propone un clasificador (ONC), el cual es una variante de FN, en la que se modifica la regla de poda FN. Con esta modificación, la regla de poda puede ser aplicada a un nodo p, sin compararlo directamente contra Q, logrando omitir más comparaciones. Esta regla es la siguiente:

$$2 * dist(Q, current_{NN}) < dist(f, current_{NN}) \quad (4)$$

Donde *f* es el prototipo más cercano al actual $current_{NN}$, contenido en el nodo p. Para reconocer al prototipo *f*, se requiere calcular y almacenar, durante una etapa de preprocesamiento, las distancias entre cada prototipo del conjunto de entrenamiento y su vecino más cercano en cada nodo del árbol. En la Figura 5 se muestra un ejemplo, en el cual se cumple la desigualdad (4), y por lo tanto se puede eliminar el nodo p.

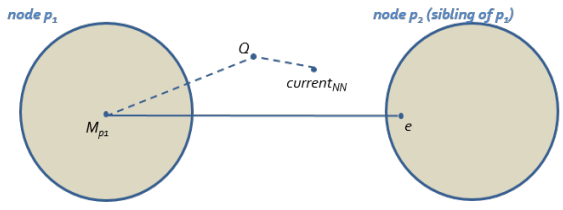

Figura 4. Interpretación gráfica del criterio de poda (GR) utilizado en Gómez-Ballester et al. (2006)

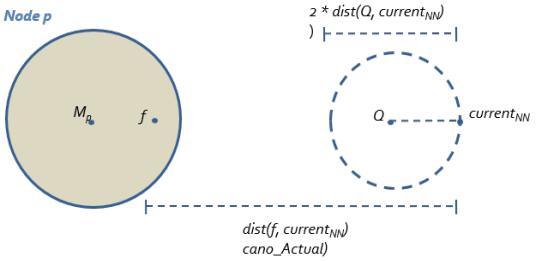

Figura 5. Interpretación gráfica del criterio de eliminación utilizado en Oncina et al. (2007)

Los clasificadores Fast k-NN mencionados hasta el momento son métodos exactos porque siempre encuentran los mismos k vecinos más cercanos que se encontrarían utilizando la búsqueda exhaustiva. Sin embargo, en algunos otros trabajos (Arya et al., 1998 y Moreno-Seco et al., 2003) se han propuesto clasificadores k-NN rápidos aproximados, basados en estructuras de árboles. Un clasificador aproximado no garantiza encontrar los k vecinos exactos más cercanos, pero se obtiene una aproximación de forma más rápida.

El clasificador propuesto en Moreno-Seco et al. (2003) utiliza el enfoque propuesto en Arya y Mount, (1998) para encontrar un (1+e) vecino más cercano, el cual permite terminar la búsqueda más rápido que en Fukunaga y Narendra (1975), pero incrementando el error en la clasificación. El algoritmo de clasificación es igual que en Fukunaga y Narendra, con la siguiente modificación en la primera regla de poda (9):

$$(1+e) * (\mathrm{dist}(Q, M_p) - R_p) < B \qquad (5)$$

Donde, e es un margen de error que permite disminuir el número de comparaciones de prototipos. Si el valor de e es cercano a cero, se obtiene la misma calidad de clasificación que se obtendría utilizando el clasificador FN, pero también se tiende a realizar el mismo número de comparaciones. Si se aumenta el valor del error e, se reduce el número de comparaciones, pero también se reduce la calidad de la clasificación (en comparación con la precisión obtenida por el clasificador FN).

2.5. Fast k-NN basados en los pasos de aproximación-eliminación

En la fase de pre-procesamiento, este tipo de clasificadores rápidos k-NN crean una matriz de distancias entre los pares de prototipos en el conjunto de entrenamiento. En la fase de clasificación, los pasos de aproximación y eliminación (basados en la desigualdad triangular) se aplican repetidamente hasta que el conjunto de entrenamiento está vacío, reportando finalmente los k vecinos más cercanos.

El primer clasificador basado en los pasos de aproximación-eliminación fue AESA (Vidal, 1986). Dado un nuevo prototipo Q para clasificar, el paso de eliminación de AESA utiliza una función de límite inferior (Lower Bound, LB), que se define de la siguiente manera: sea U el conjunto de prototipos en el conjunto de entrenamiento cuya distancia al prototipo de consulta Q ya se ha calculado, p cualquier prototipo en el conjunto de entrenamiento, aún no comparado con Q y currentNN el vecino actual más cercano de Q encontrado hasta el momento. Entonces, con base en la desigualdad triangular, se cumple que:

$$|\mathrm{dist}(Q,u) - \mathrm{dist}(u,p)| \leq \mathrm{dist}(Q,p), \quad \forall u \in U \qquad (6)$$

La distancia entre los prototipos u y p (u, pϵT) se conoce porque se almacenó en la matriz de distancias durante la fase de pre-procesamiento; y la distancia entre Q y u (u ϵ U) ya se ha calculado. En la Figura 6 se da una interpretación geométrica de la ecuación (6).

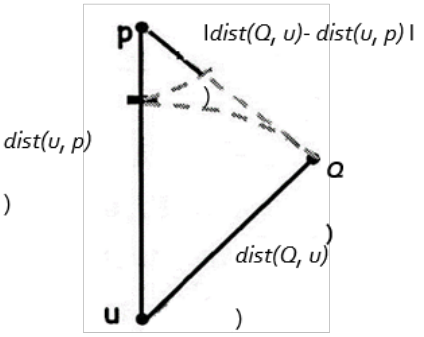

Figura 6. Interpretación gráfica del criterio de eliminación de la ecuación (12)

A partir de la Figura 6, es posible observar que la distancia real entre Q y p (dist(Q,p)) es desconocida, pero (por la propiedad de la desigualdad triangular) es menor o igual que |dist (Q,u)-dist (u,p)|. De (6), también tiene desigualdad (7):

$$\max_{\forall u \in U} |\mathrm{dist}(Q,u) - \mathrm{dist}(u,p)| \leq \mathrm{dist}(Q,p) \quad (7)$$

Así, la función LB(p), que es una estimación de la distancia real entre p y Q, se define de la siguiente manera:

$$LB(p) = \max_{\forall u \in U} |\mathrm{dist}(Q,u) - \mathrm{dist}(u,p)|, \quad \forall p \in T \quad (8)$$

La desigualdad (7) se representa gráficamente en la Figura 7. En este caso, LB(p) se selecciona como el máximo entre |dist (Q,u)-dist (u,p)| y |dist (Q,u_2)-dist (u_2,p)|, ya que el máximo es el más cercano a la distancia real entre Q y p (esto se puede notar en la Figura 7). Así, en la etapa de eliminación, todos los prototipos (pϵT) que cumplen desigualdad (9) se eliminan del conjunto de entrenamiento como posibles vecinos más cercanos.

$$LB(p) \geq \mathrm{dist}(Q, \mathrm{current}_{NN}) \quad (9)$$

En el paso de aproximación, un prototipo en T se selecciona aleatoriamente la primera vez y después de eso, dado que el límite inferior (LB) se aproxima a la distancia real, se elige el prototipo con menor valor de LB para ser comparado con Q, como el elemento más prometedor (ver Figura 8). En el ejemplo que se muestra en la Figura 8, se seleccionaría el elemento X2 ya que es el que tiene un valor LB más bajo. Los pasos de eliminación y aproximación se repiten hasta que el conjunto de entrenamiento está vacío.

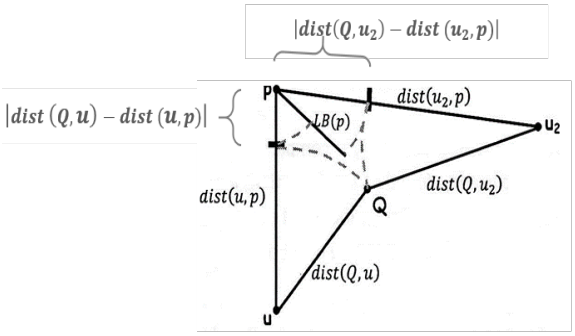

Figura 7. Interpretación gráfica de la función LB de un prototipo p

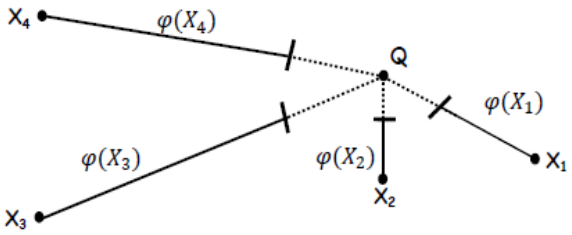

Figura 8. Interpretación gráfica de la función Lower Bound (LB), la cual aproxima la distancia real entre X1, X2, X3 y X4 con el prototipo a clasificar Q

Utilizando AESA se han obtenido buenos resultados (es decir, se reduce considerablemente el número de comparaciones). Sin embargo, el

algoritmo AESA requiere almacenar una matriz triangular que contiene todas las comparaciones entre prototipos del conjunto de entrenamiento, lo cual dificulta trabajar con conjuntos de entrenamiento grandes. Por esta razón, en Mico et al. (1995) se propone LAESA, el cual requiere seleccionar un subconjunto de prototipos de T, llamado conjunto de prototipos base BP. Posteriormente, se almacena una matriz de distancias entre los prototipos en T y los prototipos en BP, la cual es mucho más pequeña que la matriz utilizada en AESA, ya que |BP|<<|T|.

3. CONCLUSIONES

El algoritmo de clasificación del vecino más cercano presentado por Cover y Hart en 1967 ha sido muy utilizado por su sencillez y buen funcionamiento. Con las mejoras tecnológicas y algorítmicas actuales, la cantidad de información disponible en algunos campos como imágenes médicas, datos históricos bancarios, información de comercio electrónico, entre otros, ha llevado a la necesidad de crear algoritmos para la gestión de grandes conjuntos de datos. En el caso particular de la tarea de clasificación, en los últimos años muchos clasificadores rápidos k-NN para la descripción numérica de datos se han centrado en la mejora del algoritmo k-NN [1] para gestionar grandes conjuntos de datos. En este trabajo, se presentó un resumen y clasificación de algunos de estos clasificadores rápidos k-NN. Finalmente, podemos observar, a partir de esta revisión, que los clasificadores rápidos k-NN son un área de investigación activa, que vale la pena seguir explorando.

REFERENCIAS

Arya S., Mount D., Netanyahu N., Silverman R., and Wu A. (1998). "An optimal algorithm for approximate nearest neighbor searching in high dimensions". J. ACM 45 (6) 891-923.

Athitsos V., Alon J. and Sclaroff S. (2005) "Efficient nearest neighbour classification using cascade of approximate with similarity measures". Boston University Computer Science Tech. Report No. 2005-009.

Bandyopadhyay S. & Maulik U. (2002)."Efficient prototype reordering in nearest neighbor classification". Pattern Recognition 35 pp: 2791-2799.

Cheng D., Gersho A., Ramamurthi B. and Shoham Y. (1984). "Fast search algorithms for vector quantization and pattern matching". Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 1, pp. 9.11.1-9.11.4.

Cover T. M. and Hart P. E. (1967). "Nearest neighbor pattern classification". Trans. Information Theory. 13, pp. 21¬27.

Deerwester S., Dumals S., Furnas T., Landauer G. & Harshman R. (1990). "Indexing by latent semantic analysis". J. Amer. Soc. Inform. Sci. 41, pp. 391-407.

Fisher F. & Patrick E. (1970)."A preprocessing algorithm for nearest neighbour decision rules". Proc. Nat. Electronics Conf., vol. 26, pp. 481-485.

Friedman J. H., Bentley J. L. and Finkel R. A. (1977). "An algorithm for finding best matches in logarithmic expected time". ACM Transactions on Mathematical Software 3 (3), pp. 209-226.

Fukunaga K. and Narendra P. (1975). "A branch and bound algorithm for computing k-nearest neighbors". IEEE Trans. Comput. 24, pp. 743-750.

Gersho A. and Gray R. (1991). "Vector Quantization and Signal Compression". Kluwer Academic, Boston, MA.

Gómez-Ballester E., Mico L., and Oncina J. (2006). "Some approaches to improve tree-based nearest neighbor search algorithms". Pattern Recognition Letters, vol. 39 pp. 171-179.

Hwang W. & Wen K. (2002). "Fast kNN classification algorithm based on partial distance search". Electronics Letters,vol 34: 21, pp. 2062-2063.

Kalantari I. and McDonald G. (1983)."A data structure and an algorithm for the nearest point problem". IEEE Trans. Software Eng. 9, pp. 631-634.

Ke Li and Jitendra Malik. (2017). “Fast k-Nearest Neighbors via Prioritized DCI”.Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70.

K. He, F. Wen, J. Sun. (2013). "K-means hashing: An affinity-preserving quantization method for learning binary compact codes", IEEE Conference on Computer Vision and Pattern Recognition, pp. 2938-2945.

Li, Ke and Malik, Jitendra. (2016). Fast k-nearest neighbour search via Dynamic Continuous Indexing. International Conference on Machine Learning, pp. 671–679.

Lucas J. P., Luz N., and Moreno M. N. (2013). “A hybrid recommendation approach for a tourism system”. Expert Systems with Applications. 40, pp. 3532-3550.

Mico L., Oncina J., and Vidal E. (1994). "A new version of the nearest-neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing-time and memory requirements". Pattern Recognition Letters, vol. 15, pp. 9-17.

Moreno-Seco F., Mico L., and Oncina J. (2003). "Approximate Nearest Neighbor Search with the Fukunaga and Narendra Algorithm and its Application to Chromosome Classification". CIARP, LNCS 2905, pp. 322-328.

Nene S. and Nayar S. (1997). “A simple algorithm for nearest neighbour search in high dimensions”. IEEE Trans. Pattern Anal. Mach. Intell. 19 (9) 989-1003.

Niijima, S. and Kuhara, S. (2005). “Effective nearest neighbors methods for multiclass cancer classification using microarray data”. Poster presented at the 16th International Conference on Genome Informatics, December 19-21, 2005, Yokohama Pacifico, Japan.

Oncina J., Thollard F., Gomez-Ballester E., Mica L., and Moreno-Seco F. (2007). "A Tabular Pruning Rule in Tree-Based Fast Nearest Neighbor Search Algorithms". IbPRIA, LNCS 4478, pp. 306-313.

Orchard, Michael T. (1991). A fast nearest-neighbor search algorithm. In 1991 International Conference on Acoustics, Speech and Signal Processing, 1991. ICASSP-91., Volume 4 (pp. 2297–3000). IEEE Press.

Stelios and Bakamidis. (1993). “An exact fast nearest neighbour identification technique”. In IEEE International Conference on Acoustics, Speech and Signal processing. 5, pp. 658-661.

Vidal E. (1986). An algorithm for finding nearest neighbours in (approximately) constant average

time complexity. *Pattern Recognition Letters*, vol. 4, pp. 145-157.

Wolfson H. (1990). Model-Based object recognition by geometric hashing. *Proc. First European Conf. Comp. Vision*, pp. 526-536.

Yalin Chen, Zhiyang Li, Jia Shi, Zhaobin Liu, Wenyu Qu. (2018). *Stacked K-Means Hashing Quantization for Nearest Neighbor Search*. 2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM).

Yunck, T. (1976). *A technique to identify nearest neighbors*. IEE Trans. On S.M.C., SMC-6:10.