IIC2440 – Procesamiento de Datos Masivos

Locally-Sensitive Hashing

1. Motivacion y definiciones

La técnica de Locally-sensitive Hashing intenta resolver el siguiente problema.

> Dada una colección de elementos y una noción de distancia, retornar los pares de elementos más cercanos de acuerdo a esa distancia.

Y la forma de resolverlo, es usando funciones o métodos de hashing, que cumplan con la siguiente propiedad: los hashes de elementos más cercanos tienen una probabilidad alta de ser iguales, y los hashes de elementos lejanos tienen una probabilidad baja de ser iguales.

Recordemos que una distancia es una función d sobre pares de elementos de un conjunto, que cumple con las siguientes propiedades:

- d es simétrica.
- $d(x, x) = 0$
- $d(x, y)$ es siempre positiva
- Satisface la desigualdad triangular: $d(x, z) \leq d(x, y) + d(y, z)$

Para efectos de esta definición suponemos un conjunto E de elementos, equipado con una distancia d.

Definición. Una familia F de funciones $E \to \mathbb{R}$ es (d_1, d_2, p_1, p_2)-sensitiva si para cada par de elementos x e y, y una función $f \in F$ tomada aleatoriamente, tenemos que:

- Si $d(x, y) \leq d_1$, entonces $f(x) = f(y)$ con probabilidad mayor o igual a p_1
- Si $d(x, y) \geq d_2$, entonces $f(x) = f(y)$ con probabilidad menor o igual a p_2

¿Para que sirve esto? Podemos usar la siguiente receta para (intentar) tomar todos aquellos pares de elementos tales que $d(x, y) \geq k$:

- calculamos $f(e)$ para cada elemento $e \in E$.
- para cada par de elementos (x, y) tal que $f(x) = f(y)$, calculamos $d(x, y)$. Si es mayor o igual a k, lo reportamos.

Si queremos reportar pares de elementos tales que $d(x, y) \leq k$ y estamos dispuestos a tolerar que podamos tener falsos negativos, siempre y cuando la probabilidad de no tomar un par (x, y) con $d(x, y) \leq k$ sea menor o igual a p_{fp}, entonces necesitamos una familia de funciones que sea $(k, k', 1 - p_{fp}, p')$. En este ejemplo, el valor de k' y p' nos permite modular cuan costosa es nuestra búsqueda: mientras más pequeña es k' y más pequeña es p', vamos a tener menos pares donde $f(x) = f(y)$.

2. MinHashing

El ejemplo de un esquema Locally-sensitive hashing que veremos en este curso tiene las siguientes componentes:

- Los elementos del conjunto E serán conjuntos (por ejemplo, el resultado de hacer shingling a texto).

- La métrica de distancia entre un conjunto x y un conjunto y corresponde a Jaccard: es 1 menos la similitud de Jaccard entre x e y, es decir, $(1 - \frac{|x \cap y|}{|x \cup y|})$ (asi, nos queda que la distancia de jaccard entre un elemento y si mismo es 0).

La familia de funciones F estará dada por las llamadas funciones de MinHashing, que definimos de la siguiente forma.

Supongamos que tenemos un conjunto E de conjunctos, es decir, $E = C_1, C_2, \ldots, C_n$, y que el universo $\bigcup C_i = e_1, \ldots, e_m$ tiene m elementos. Podemos escribir E en forma matricial, con m filas y n columnas, donde $E[i, j] = 1$ si el elemento e_j está en C_i, y 0 en otro caso.

La familia F de funciones de MinHash está dada en este caso por cada una de las $m!$ permutaciones de las filas de la matriz E. Para cada una de esas permutaciones π, la función de minhash $f_\pi : \{0, \ldots, n-1\} \to \{0, \ldots, n-1\}$ asociada a π se define de la siguiente forma: $f(i)$ es el el numero correspondiente a la primera fila en $\pi(E)$ donde hay un 1 en la columna C_i.

Observación 1. Para dos conjuntos x, y y una permutación al azar, la probabilidad que $f_\pi(x) = f_\pi(y)$ corresponde a la similitud de Jaccard entre x e y (es decir, a $\frac{|x \cap y|}{|x \cup y|}$).

Para demostrar esto, concentremonos solo en las columnas correspondientes a x e y en la matriz E. Las filas de esta submatriz son de tres tipos: (1) tienen un 0 en ambas columnas, o (2) tienen un 1 en ambas columnas, o (3) una columna tiene un 1 y otra tiene un 0.

Notemos que la similitud de jaccard corresponde entonces a la cantidad de las filas de tipo 2, dividido por la cantidad de filas de tipo 2 o tipo 3 en la matriz.

Veamos entonces que pasa en una permutación π al azar de la submatriz. Si nos encontramos primero una fila de tipo 2, entonces por definición $f_\pi(x) = f_\pi(y)$. Contando, la probabilidad que esto pase es precisamente la cantidad de las filas de tipo 2, dividido por la cantidad de filas de tipo 2 o tipo 3 en la matriz. Por otro lado, si nos encontramos primero una fila de tipo 3, entonces por definición $f_\pi(x) \neq f_\pi(y)$.

Observación 2. Para cada $0 \leq d_1 \leq d_2 \leq 1$, la familia de funciones de MinHashing es $(d_1, d_2, 1 - d_1, 1 - d_2)$-sensitiva.

Para ver esto, consideremos un par de elementos (x, y) tal que su distancia de Jaccard es $d(x, y)$. Entonces

- Si $d(x, y) \leq d_1$, entonces tenemos que probar que $f(x) = f(y)$ con probabilidad mayor o igual a $1 - d_1$. Pero ya mostramos que $f(x) = f(y)$ con probabilidad igual a la similitud de Jaccard, $1 - d(x, y)$, y como $d(x, y) \leq d_1$, $1 - d(x, y) \geq 1 - d_1$, que es lo que buscamos.

- El caso cuando $d(x, y) \geq d_2$ se demuestra de forma analoga.

Conclusión. Perfecto! tenemos entonces nuestro esquema locally sensitive. El problema es que no sirve de mucho. Supongamos que queremos todos los elementos con similitud de Jaccard 0,8, o distancia de Jaccard 0,2. La propiedad de arriba nos dice que los elementos asi de cercanos serán mapeados con probabilidad 0,8. Esto igual deja un 20 % de los elementos fuera, ¡es demasiado!

3. Amplificando familias de funciones sensitivas

Para magnificar las probabilidades, de forma que nos queden mas razonables, vamos a utilizar de forma paralela estas dos técnicas sobre una familia F de funciones sensitivas. Notemos que las funciones resultantes son funciones binarias y booleanas, solo nos van a servir para calcular si $f(x) = f(y)$ o $f(x)! = f(y)$.

OR Tomamos una banda b de funciones en F, digamos $f_1, \ldots, f_b$, y nos quedamos con la disjunción de todas ellas. Más precisamente, la nueva familia de funciones F^{OR} contiene una función por cada conjunto de b funciones en f. Esa función f^{OR} se define como $f^{\text{OR}}(x, y) = 1$ si existe algún $f_i \in \{f_1, \ldots, f_b\}$ tal que $f_i(x) = f_i(y)$, y $f^{\text{OR}}(x, y) = 0$ en otro caso. Si la familia original era (d_1, d_2, p_1, p_2)-sensitiva, entonces la familia F^{OR} es $(d_1, d_2, 1 - (1 - p_1)^b, 1 - (1 - p_2)^b)$-sensitiva

AND Tomamos una conjunto r de funciones en F, digamos $f_1, \ldots, f_r$, y nos quedamos con la conjunción de todas ellas. Más precisamente, la nueva familia de funciones F^{AND} contiene una función por cada conjunto de b funciones en f. Esa función f^{AND} se define como $f^{\text{AND}}(x, y) = 1$ si para todo $f_i \in \{f_1, \ldots, f_b\}$ se tiene que $f_i(x) = f_i(y)$, y $f^{\text{AND}}(x, y) = 0$ en otro caso. Si la familia original era (d_1, d_2, p_1, p_2)-sensitiva, entonces la familia F^{AND} es (d_1, d_2, p_1^r, p_2^r)-sensitiva

Evidentemente, tomar el AND hace disminuir las probabilidades en la definición de ser sensitiva, y el OR las hace aumentar. La gracia es ir intercalando estas construcciones para bajar p_2 y subir p_1.

Amplificando funciones de MinHashing. Para nuestro ejemplo con Jaccard, el esquema es el siguiente.

Tomaremos b bandas o conjuntos de funciones, cada una con r funciones de minhashing (dadas por r permutaciones distintas). Para dos conjuntos x e y, vamos a decir que el LSH de x e y los asigna como *candidatos a ser similares*, que escribiremos como $LSH_{b,r}(x, y) = 1$, si para alguna de las b bandas se tiene que $f(X) = f(y)$ en cada una de las r funciones.

Consideremos que la similitud de Jaccard de x e y es s (y por tanto la distancia de Jaccard es $1 - s$). Entonces la probabilidad que $LSH_{b,r}(x, y) = 1$ corresponde a $1 - (1 - s^r)^b$.

Actividad Propuesta. Para distintos valores de b y r, experimenta graficando esta función. recordemos que el esquema LSH es $(1 - s, 1 - s, 1 - (1 - s^r)^b, 1 - (1 - s^r)^b)$-sensitivo. ¿Cuáles son buenos valores de b y r?