ANALISI DI DATI SU LARGA SCALA

Prof. Dario Malchioni
6 CFU

Luca Cappelletti

Lecture Notes
Year 2017/2018

Magistrale Informatica
Università di Milano
Italy
2 novembre 2017

Indice

Capitolo 1

Introduzione

1.1 Data Mining

È definito come un insieme di tecniche per la raccolta di modelli per dati. In **statistica** è stato utilizzato per realizzare *modelli statistici* cercando pattern tra grandi quantità di dati, e ovviamente è in uso nella ricerca del **machine learning** dove vengono utilizzati sia per allenare i network sia per ricercare nuovi dati. Una delle potenziali applicazioni del data mining è la **summarization**, cioè quando si estrapolano dati rappresentativi di altri grossi gruppi, come nel caso di *PageRank*.

1.1.1 Limiti del data mining: il principio di Bonferroni

Il principio di Bonferroni afferma che, una volta calcolato il numero atteso delle occorrenze degli eventi che si stanno cercando, considerando i dati randomici,

Capitolo 2

Jobs

Nelle applicazioni moderne del data-mining vengono effettuati calcoli che necessitano un grande potere computazionale. Questo non viene computato da un super-computer, ma da un cluster di computer, realizzato con un **file system distribuito** ed un sistema di programmazione chiamato **MapReduce**.

2.1 Distributed file system (DFS)

Vengono realizzati su cluster di computer, in cui i file son ridondati e le computazioni divise in tasks, in modo tale da prevenire possibili errori. Esempi di DFS sono il *Google File System (GFS)* e *Hadoop*. I file tendenzialmente sono enormi, anche TB di dimensione e sono raramente aggiornati.

2.1.1 La funzione Map

È una funzione scritta dall'utente che, data una collezione di oggetti in input, converte ognuno in una o più coppie chiave-valore (key-value). Le chiavi in questo caso non sono necessariamente uniche.

2.1.2 La funzione Reduce

Il sistema MapReduce si occupa di suddividere le coppie key-value prodotte dai Map tasks ai Reduce tasks. Ogni Reduce task combina gli elementi su ogni lista, applicandovi la funzione scritta dall'utente. Il risultato prodotto forma l'output del processo MapReduce.

2.2 Analisi di complessità di un job map-reduce

2.2.1 Es: moltiplicazione di matrici

$$A_{m\times n}\times B_{m\times o}(i,j,a_{ij})\longmapsto M_A((i,j),(A,k,a_{ik}))\forall\, j=1,...,o(k,j,b_{kj})\longmapsto M_B((i,j),(B,k,b_{ik}))\forall\, i=1,...,m(i,j)[(A,1,a_{i1}),...,(A,m,a_{im})$$

$$R(A,B)\bowtie S(B,C)\bowtie T(C,D)(a,b)\longmapsto_{M_R}(b,(R,a))(b,c)\longmapsto_{M_S}(b,(S,a))$$

Quale è il costo di questo algoritmo. Indichiamo con r,s,t i rispettivi numeri di tuple delle tabelle R, S, T. Il primo processo M_R riceve tutte e sole le tuple di R, quindi ha costo r. Il secondo, similmente, ha costo s.

Il risultato del costo di complessità sarà quindi un $O(r+s)$. Ma questo è tra due relazioni. Se volessi farlo da 3 relazioni (**join in cascata**) cosa andrei ad ottenere?

$(R\bowtie S)\bowtie T$

Ottengo il costo $O(r+s+t+rsp)$, con p rappresentante la probabilità che due valori di R e S hanno un attributo uguale.

2.2.2 Join multi-way

Date due funzioni di hash, una h per l'attributo B ed una g per l'attributo C, con b **bucket** e c **bucket**, avendo che $bc = k$.
Nel caso di una tupla $(u,v) \in R$ viene inviata ad un'unica colonna verticale, riducendo i nodi (**c reducer**).
Nel caso di una tupla $(w,z) \in T$ viene inviata ad un'unica colonna orizzontale, riducendo i nodi (**b reducer**).
Nel caso di una tupla $(v,w) \in S$ viene inviata ad un'unica cella, riducendo i nodi ad uno soltanto (**1 reducer**).
Il costo quindi risulta essere:

$$O(r+2s+t+cr+bt)$$

2.2.3 Rilassamento lagrangiano

N.B. Il parametro lambda non può essere negativo.

$$L(b,c) = cr + br - \lambda(bc-k)$$

$$\frac{dL(b,c)}{db} = 0$$

$$\frac{dL(b,c)}{dc} = 0$$

Ottengo quindi un sistema:

$$\begin{cases} t - \lambda c = 0 \\ r - \lambda b = 0 \end{cases} \implies \begin{cases} t = \lambda c \\ r = \lambda b \end{cases}$$

$$\lambda = \sqrt{\frac{rt}{k}}$$

$$c = \sqrt{\frac{kt}{r}}$$

$$b = \sqrt{\frac{kr}{t}}$$

Il **costo ottimizzato** della **join multiway** risulta quindi essere:

$$O(r+2s+t+2\sqrt{krt})$$

2.2.4 Es: Join sui nodi di facebook

Prendiamo ad esempio il grafo dei nodi facebook, dotato di 10^9 nodi.

$R(U_1,U_2)$, $|R| = r = 3 \times 10^1 1$ (dati arbitrari)

$$R \bowtie R \bowtie R$$

Approccio Multi-way: $r+2r+r+2r\sqrt{k} = 4r+2r\sqrt{k} = 1 \times 2 \times 10^1 2 + 6 \times 10^1 1\sqrt{k}$
Approccio cascata (nell'ipotesi che $absR \bowtie R = 30r$): $r+r+r+r^2 \times p = ... = 2r+60r = 1 \times 2 \times 10^1 2 + 1 \times 86 \times 10^1 3$

Ottengo quindi che: $6 \times 10^1 1\sqrt{k} \leq 1 \times 86 \times 10^3 3 \longrightarrow k \leq 961$, e risulta quindi migliore utilizzare l'approccio multi way quando si hanno meno di 961 nodi da allocare a dei reducer.

2.2.5 Es: Google pagerank

Come funziona **pagerank**:

	A	B	C	D
A	0	1/2	1	0
B	1/3	0	0	1/2
C	1/3	0	0	1/2
D	1/3	1/2	0	0

$$v_j(t+1) = P(\text{Trovarsi in j al tempo t+1}) = \sum_i P(\text{trovarsi in i al tempo t}) \bullet P(\text{spostarsi da i a j} \mid \text{trovarsi in i al punto t})$$

$$\sum_i v_i(t) m_{ji} = \sum_i m_{ji} v_i(t) = (Mv(t))_j$$

con $M_{ij} = P$Spostarsi da j a i.

$$\vec{V}(t+1) = M\vec{v}(t)$$

Prova di convergenza

Definizione 2.2.1 (Determinante)

$$detA = \sum_i a_{ij} c_{ij} = \sum_j a_{ij} c_{ij}$$

Definizione 2.2.2 (Determinante di matrice trasposta) *Una matrice e la sua trasposta possiedono lo stesso determinante.*

$$detA^T = \sum_i a^T_{ij} c^T_{ij} = \sum_i a_{ji} c_{ji} = \sum_j a_{ij} c_{ij}$$

Definizione 2.2.3 (Autovalori) *Si ottengono trovando gli zeri dell'equazione caratteristica.*

$$det(A - \lambda I) = 0 \leftrightarrow det(A - \lambda I)^T = 0 \leftrightarrow (A^T - \lambda I) = 0$$

Definizione 2.2.4 (Matrice stocastica per righe) *Una qualsiasi matrice stocastica per righe ammette* 1 *come autovettore.*

$$\forall i \sum_j a_{ij} = 1$$

Definizione 2.2.5 (Matrice stocastica per colonne) *Una qualsiasi matrice stocastica per colonne ammette* 1 *come autovettore poiché si tratta della trasposta di una matrice stocastica per righe.*

Teorema 2.2.6 (La potenza di una matrice stocastica è sempre stocastica) *Il risultato dell'elevazione a potenza di una matrice stocastica risulta sempre essere stocastico. Dimostriamo per induzione:*

$$\textit{Base: } k = 1 \qquad A^k = A$$

$$\textit{Passo: } A^k \textit{ stocastica} \rightarrow A^{k+1} \textit{ stocastica}$$

Dimostriamo che ad un generico passo k, otteniamo sempre 1 quando andiamo a sommare i termini.

$$a^{k+1}_{ij} = \sum_s a^k_{ij} a_{sj} \sum_j a^k_{ij} = \sum_j \sum_s a^k_{is} a_{sj}$$

Inverto le sommatorie ed ottengo:

$$\sum_s a^k_{is} \sum_j a_{sj} = \sum_s a^k_{is} = 1$$

Teorema 2.2.7 (Autovalori di una stocastica) *Se A è stocastica per colonne, il suo autovalore massimo è 1.*

Procediamo a dimostrare per assurdo.

Sappiamo che 1 è un autovale di A, poiché A è stocastica. Affermiamo che, per assurdo, esista un $\lambda > 1$ autovalore di A (che è anche l'autovalore di A^T) e v un autovettore per λ.

$$A^T v \Rightarrow \lambda v$$

$$A^{2T} v = A^T \times A^T v = A(\lambda v) = \lambda A^T v = \lambda^2 v \Rightarrow A^{Tk} v = \lambda^k v$$

Maggioriamo / minoriamo la sommatoria:

$$\sum_j a_{ij}^{Tk} v_{max} \geq \sum_j a_{ij}^{Tk} v_j = \lambda^k v_i > G$$

$$v_{max} \sum_j a_{ij}^{Tk} > G$$

Ma siccome la sommatoria è dei termini di una matrice stocastica per righe (essendo la trasposta di A), deve essere pari a 1, per cui:

$$v_{max} \sum_j a_{ij}^{Tk} = v_{max} > G$$

$$1 > \frac{G}{v_{max}} \Rightarrow \textit{assurdo!}$$

Teorema 2.2.8 (Autovalori di potenza di matrice)

$$v_0 \to v_1 = Av_0 \to v_2 = Av_1 = A^2 v_0 \to \ldots \to v_k = A^k v_0$$

Dimostrazione

$$Av_0 = A(\alpha_1 x_1 + \alpha_2 x_2 + \ldots + \alpha_n x_n) \qquad = \alpha_1 Ax_1 + \alpha_2 Ax_2 + \ldots + \alpha_n Ax_n = \alpha_1 \lambda_1 x_1 + \alpha_2 \lambda_2 x_2 + \ldots + \alpha_n \lambda_n x_n$$

$$v_k = A^k v_0 = \alpha_1 \lambda_1^k x_1 + \alpha_2 x_2 \frac{\lambda_2^k \lambda_1^k}{\lambda_1^k} + \ldots + \alpha_k x_k \frac{\lambda_k^k \lambda_1^k}{\lambda_1^k}$$

$$v_k = A^k v_0 = \lambda_1^k (\alpha_1 x_1 + \alpha_2 x_2 \frac{\lambda_2^k}{\lambda_1^k} + \ldots + \alpha_k x_k \frac{\lambda_k^k}{\lambda_1^k})$$

per k "grande" $v_k = A^k v_0 \approx \lambda_1^k \alpha_1 x_1$, nel nostro caso $\lambda_1 = 1$, cioè $\lim_{k \to \infty} \alpha_1 \lambda_1^k x_1 = 1$.

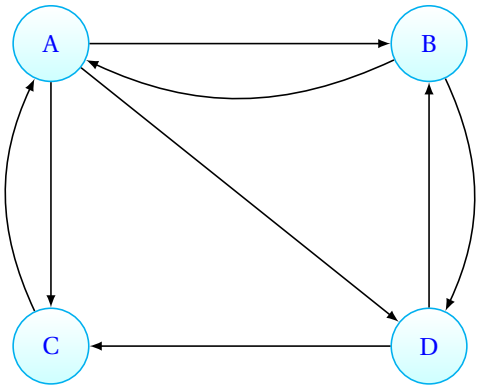

Figura 2.1: In questo grafo, dopo 10 esecuzioni, pagerank assegna assegna ai nodi A, B, C, D rispettivamente $^3/_9, ^2/_9, ^2/_9 e ^2/_9$.

2.2.6 Trappole per ragni

Nodi di grafi (figura 2.2) con loop, da cui gli spider non possono uscire.

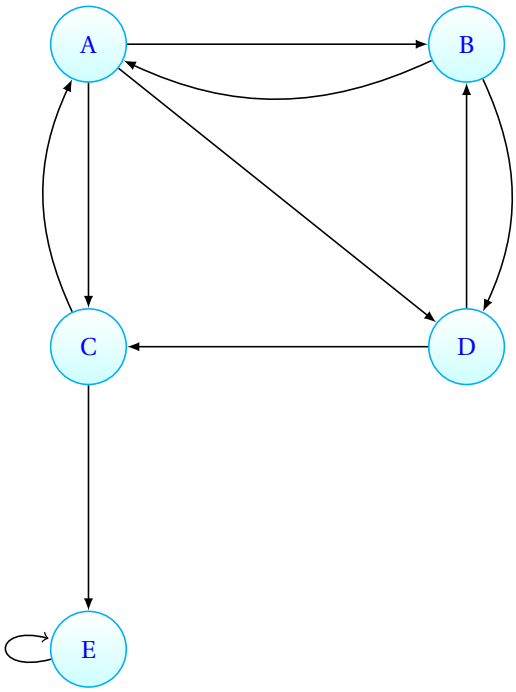

Figura 2.2: Dopo un certo numero di iterazioni, tutta la "massa" distribuita inizialmente sul grafo sarà finita su E

.

2.2.7 Risolvere le trappole per ragni col teletrasporto

Per evitare le trappole per ragni, andiamo a inserire una certa probabilità β di teletrasportarci da un nodo ad un altro al posto di continuare a navigare tramite link.

$$false_{t+1} = \beta M false_t + (1-\beta)\frac{1}{n} false$$

P(mi trovo in i al tempo $t+1$) = P(spostarsi tramite link in i | scelgo i link) P(scelgo i link) + P (teletrasportarsi in i | mi teletrasporto) P(scegliere teletrasporto)

Capitolo 3

Spark

Spark è un motore generale e veloce per il processo di dati su larga scala che disaccoppia completamente il concetto di storage da quello di calcolo distribuito, consentendo di avere software esterni per il file system distribuito come Hadoop.

3.1 Resilient Distributed Dataced RDD

Ho la stessa garanzia di **Hadoop** per quanto riguarda la computazione, partendo da file distribuiti su un cluster sulla home/RDD.

3.1.1 Trasformazioni

Una funzione che prende un dataset e lo modifica. Esempi di questa categoria sono **map**, **filter**, **flatMap** e **groupByKey**.

3.1.2 Actions

Una funzione che compie un'azione sul dataset. Esempi di questa categoria sono **reduce**, **count**, **collect** e **take**.

3.1.3 Variabile broadcast

Variabile a sola lettura condivisa tra tutti i nodi.

3.1.4 Accumulatori

Un accumulatore può essere usato solo per operazioni strettamente associative, come conteggi.

3.2 Metodo di tassazione

Fino ad ora lo abbiamo visto come risoluzione per i problemi di convergenza dell'algoritmo di pagerank, ma è utilizzabile anche in altri casi come la personalizzazione del risultato del vettore prodotto, e viene utilizzato in algoritmi come **topic-sensitive pagerank**, dove per esempio se volessi personalizzare i risultati di un utente appassionato di sport farei teletrasportare il crawler a pagine sicure che so che parlano di sport.

3.2.1 Pagerank vs Trustrank

Se chiamiamo pagerank p e trustrank t, proviamo a calcolare $\frac{p-t}{p}$. Se questa quantità è negativa o vicina a zero, sono contento, se invece si avvicina all'uno probabilmente su quel nodo sono attive delle tecniche che falsano il pagerank e riducono il Trustrank.

3.2.2 HITS, chiamato anche Hub&Authorities

Definisce le pagine "buone" come fa pagerank, ma utilizza una ricorsione indiretta. Divide tutte le pagine web in due categorie, **hub** e **authorities**. Una pagina è **hub** quando linka delle buone authorities, mentre chiamata **authority** quando è linkata da un buon hub.

Capitolo 4

Similarità

Definizione 4.0.1 (Indice di similarità di insiemi di Jaccard) *Dati due insiemi A e B, la similarità dei due insiemi sarà definita come:*

$$SIM(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Es: due utenti sono definibili simile dall'insieme di oggetti che essi hanno acquistato.

Definizione 4.0.2 (K-gramma (Shingling)) *Stringa di k caratteri che appare consecutivamente in un documento. Se volessimo rappresentare un documento tramite il suo k-gramma, il suo indice di Jaggard del k-gramma misura la similarità testuale dei documenti.*

Definizione 4.0.3 (Stop word) *Parole comuni nel linguaggio naturale ma che non aggiungono particolare valore semantico ad un testo.*

In alcuni contesti vengono tolti gli spazi nei documenti per calcolare il k-gramma di un documento, ma questo in alcuni casi può far perdere informazioni sul documento (Es: "Touch down" nel contesto dell'atterraggio di un aereo o di una partita di rugby).

Definizione 4.0.4 (Matrice rappresentativa di un Insieme) *Le colonne della matrice corrispondono agli insiemi, mentre le righe corrispondono agli elementi del set universale da cui i set sono estratti. Viene posto un 1 nella cella sulla riga r e colonna c se l'elemento r è un membro del set c, altrimenti è 0.*

Elemento	S_1	S_2	S_3	S_4
a	1	0	0	1
b	0	0	1	0
c	0	1	0	1
d	1	0	1	1
e	0	0	1	0

Figura 4.1: Nella matrice rappresentativa troviamo $\Delta = \{a,b,c,d,e\}, S_1 = \{a,d\}, S_2 = \{c\}, S_3 = \{b,d,e\}, S_4 = \{a,c,d\}$.

4.1 Minhash

Definizione 4.1.1 (Minhash) *Il valore di minhash di una qualsiasi colonna è il primo numero nella prima colonna, nella data permutazione (le righe della matrice possono essere permutate) che ha come valore 1. Ogni calcolo minhash va a costruire la firma di un set, che è composta da un grande numero di questi calcoli.*

4.1.1 Connessione tra minhashing e indice di Jaccard

La probabilità che una funzione di minhash per una permutazione randomica di righe produca lo stesso valore per due insiemi **è uguale** alla similarità di Jaccard per questi due insiemi.

4.1.2 Firma di minhash