Μάθημα: Συστήματα Διαχείρισης Δεδομένων Μεγάλης Κλίμακας
Διδάσκων : Δημήτρης Μιχαήλ
Ακ. Έτος :2015-2016

Αριθμός Ομάδας: 143
Μέλη Ομάδας: 15103- Κόλιας Γεώργιος
15402-Λοντόρφος Νίκος

Υπολογισμός tf-idf με Hadoop

Ορισμός

Στην ανάκτηση πληροφοριών το βάρος TF-IDF, είναι ένα αριθμητικό στατιστικό στοιχείο που αντανακλά πόσο σημαντική είναι μια λέξη σε ένα έγγραφο που ανήκει σε μια συλλογή εγγράφων και χρησιμοποιείται συχνά ως ένας συντελεστής στάθμισης. Η τιμή του TF-IDF αυξάνει αναλογικά με τον αριθμό των φορών που μια λέξη εμφανίζεται σε ένα έγγραφο, αλλά είναι αντιστρόφως ανάλογη με τη συχνότητα εμφάνισης της λέξης στο σώμα-συλλογή κειμένων.

Το βάρος $\boldsymbol{tf/idf_{t,d}}$ *ενός όρου t είναι σε ένα έγγραφο d είναι το γινόμενο:*

$$w_{t,d} = \log_{10}(1+tf_{t,d}) \times \log_{10}\left(\frac{N}{df_t}\right)$$

Υλοποίηση

Για να τρέξουμε το πρόγραμμα θα πρέπει να δώσουμε 3 ορίσματα ως είσοδο: ***[input_folder] [output folder] [number_of_docs].*** *π.χ.:*
hadoop/bin/hadoop jar /vagrant/data/tf_idf_m/tf_idf_m/target/tf_idf-0.1.jar org.hua.tf_idf.Tf_idf_m ***/tfidf_m/input /tfidf_m/output 100***

Το πρόβλημα υπολογισμού του βάρους $tf/idf_{t,d}$ *σε μια συλλογή εγγράφων υλοποιήθηκε σε 3 φάσεις χρησιμοποιώντας 3 jobs (jobtf, jobdf, jobtfidf) και αντίστοιχα 3 κλάσεις : **tf_m, df_m, tf_idf_m_final.** Η κάθε κλάση υλοποιεί ένα mapper και ένα reducer χρησιμοποιώντας το v2 API του Hadoop.*

Στην πρώτη φάση υπολογίζουμε το $tf_{t,d}$ *για κάθε όρο σε κάθε έγγραφο της συλλογή* $\boldsymbol{tf_{t,d}}$***:*** *αριθμός εμφανίσεων του όρου στο έγγραφο.*

Στη δεύτερη φάση χρησιμοποιούμε τα αποτελέσματα της 1ης φάσης ως είσοδο για να πάρουμε το κάθε όνομα αρχείου 1 φορά(distinct values) και υπολογίζουμε το $\mathbf{Df_{t,d}}$ ***:*** *ο αριθμός (πλήθος) των εγγράφων της συλλογής που περιέχουν τον όρο t.*

Στη τελική 3η φάση παίρνουμε ως είσοδο τα αποτελέσματα της 1ης και 2ης φάσης . Στη συνέχεια διαβάζουμε από το configuration τον αριθμό των εγγράφων (είναι το 3ο όρισμα εισόδου το οποίο το περνάμε στο configuration) και υπολογίζουμε τα tf/idf$_{t,d}$ *βάρη.*

1η φάση: Jobtf

Υπολογισμός : $\mathbf{tf_{t,d}}$
Κλάση: **Tf_m**

Είσοδος: Ως είσοδο έχουμε το directory της συλλογής κειμένων. Είναι το 1ο όρισμα args[1]: ***[input_folder]***.

Mapper:

Είσοδος: <document, each line contents >
Έξοδος: < <word,doc>, 1>

Περιγραφή
Στην πρώτη φάση διαβάζουμε κάθε γραμμή του κειμένου και χρησιμοποιώντας το "context.getInputSplit()).getPath().getName();" παίρνουμε το όνομα του κάθε κειμένου(**doc**).Στη συνέχεια χρησιμοποιώντας το Pattern.compile("\\w+") βρίσκουμε τις λέξεις του κειμένου (word) και για κάθε μια από αυτές γράφουμε στο context : < <word,doc>, 1>.

Reducer:

Είσοδος: < <word,doc>, 1,1,1,1,1,1>
΄Εξοδος: <<word,doc>, $tf_{t,d}$ >

Περιγραφή
Στον reducer ομαδοποιούμε τα αποτελέσματα του mapper με βάση το κλειδί: <word,doc> και προσθέτουμε τους άσσους, βγάζοντας ως τελικός αποτέλεσμα το <<word,doc>, $tf_{t,d}$>. Τα αποτελέσματα του job σώζονται στο "hdfs://localhost:54310/tfidf_m/outputTF" του συστήματος αρχείων στο virtual machine.

2η φάση :jobdf

Υπολογισμός : **$df_{t,d}$**
Κλάση **Df_m**

Mapper:

Είσοδος: <<word,doc>, $tf_{t,d}$>-Τα αποτέλεσμα του **Jobtf.**
΄Εξοδος: <<word,"dummy">, doc>

Περιγραφή
Στο mapper διαβάζουμε κάθε <<word,doc>, $tf_{t,d}$> και κάνουμε split στον κενό χαρακτήρα, ώστε να διαχωρίσουμε το κλειδί word,doc από την τιμή του $tf_{t,d}$. Στη συνέχεια γράφουμε στην έξοδο <word,"dummy"> ως κλειδί και doc ως τιμή.

Reducer:

Είσοδος<<word,"dummy">, doc1,doc2,doc3..>
Έξοδος: <<word,"dummy">, $df_{t,d}$>

Περιγραφή
Στον reducer ομαδοποιούμε τα αποτελέσματα του mapper με βάση το κλειδί: <word,"dummy"> και προσθέτουμε 1 για κάθε doc που μας έρχεται ως τιμή. Με αυτό τον τρόπο υπολογίζουμε σε πόσα έγγραφα εμφανίζεται ο κάθε όρος.

3η φάση: jobtfidf

Υπολογισμός **: tf/idf**
Κλάση **tf_idf_m_final**

Mapper:

Είσοδος: <<word,doc>, $tf_{t,d}$> ή **<<word,"dummy">, $df_{t,d}$>**
(Τα αποτελέσματα του **Jobtf** και του **jobdf**).
Έξοδος: <<word>, inputdirectory,doc1 ,$tf_{t,d}$>
<<word>, inputdirectory ,doc2 ,$tf_{t,d}$>
….
<<word>, inputdirectory,"dummy" ,$df_{t,d}$>

Περιγραφή
Στο mapper διαβάζουμε κάθε γραμμή από τα αποτελέσματα του jobtf <<word,doc>, $tf_{t,d}$> και το input directory και το μετατρέπουμε σε <<word>, inputdirectory ,doc, $tf_{t,d}$>.
Αντίστοιχα διαβάζουμε τα αποτελέσματα
του jobdf <<word,"dummy">,$df_{t,d}$> και το input directory και τα μετατρέπουμε σε <<word>, inputdirectory ,"dummy", df(t,d)>.

Reducer:

Είσοδος: <word, (inputdirectory,doc1,$tf_{t,d}$), (inputdirectory ,doc2,$tf_{t,d}$),....(inputdirectory ,dummy,$df_{t,d}$)>
΄Εξοδος: <<word,doc>, $tfidf_{t,d}$,$tf_{t,d}$,$idf_{t,d}$>

Περιγραφή
Στον reducer ομαδοποιούμε τα αποτελέσματα του mapper με βάση το κλειδί: <word> και υπολογίζουμε το $idf_{t,d}$ και τα τελικά βάρη $tf/idf_{t,d}$, χρησιμοποιώντας τον αριθμό των εγγράφων της συλλογής που διαβάζουμε από το configuration.

Συγκεκριμένα κάνουμε μια επανάληψη στη συλλογή από values που μας έρχονται για κάθε λέξη: **<(inputdirectory,doc1,$tf_{t,d}$),(inputdirectory,doc2,$tf_{t,d}$),....(inputdirectory ,dummy,$df_{t,d}$)>**.
Μέσα στην επανάληψη ελέγχουμε αν το inputdirectory περιέχει τη λέξη που έχουμε ορίσει στον configuration ώστε να καταλαβαίνουμε ότι προήλθε από το jobdf και επομένως περιέχει την τιμή df της λέξης. Την τιμή αυτή θα τη χρησιμοποιήσουμε για τον υπολογισμό του $idf_{t,d}$ της λέξης. Εάν δεν περιέχει τη λέξη πρόκειται για value της μορφής: **(inputdirectory ,doc1,$tf_{t,d}$),** το οποίο προήλθε από τα αποτελέσματα του Jobtf και σώζουμε την πληροφορία του ονόματος του εγγράφου και του $tf_{t,d}$ σε ένα HashMap.

Αφού ολοκληρωθεί η παραπάνω διαδικασία κάνουμε μια επανάληψη σε όλα τα στοιχεία του Hashmap και υπολογίζουμε το $tfidf_{t,d}$ για κάθε συνδυασμό λέξης και εγγράφου.