Tesina di
Big Data Management

Corso di Laurea Magistrale in
Ingegneria Informatica e Robotica
curriculum data science

A.A. 2021/22

docente
Fabrizio Montecchiani

SVILUPPO DI UN CLUSTER SPARK-HADOOP PER SENTIMENT ANALYSIS DI TESTI

334568	**Andrea Baldinelli**	andrea.baldinelli@studenti.unipg.it
338528	**Cristiano Bernardini**	cristiano.bernardini@studenti.unipg.it

Sommario

1. Introduzione

Lo scopo del progetto in questione è quello di un modello di machine learning che, analizzando una frase fornita dall'utente da riga di comando, sia in grado di valutare se a quest'ultima possa essere associato un sentimento positivo o negativo.
Ad esempio, se l'utente scrivesse: "*It's a very bad day*", il modello dovrebbe indicare che la frase ha valenza negativa; analogamente, se la frase fosse: "*I love Rome*" si avrebbe un risultato positivo.

Il tutto è eseguito all'interno di un ambiente distribuito, sia in termini di calcolo sia in termini di file system.

Come modello si è scelto un classificatore binario, nello specifico un ***Naive Bayes Classifier***, che, nonostante si basi sull'ipotesi, raramente verificata, che parole adiacenti siano incorrelate tra loro, è particolarmente efficace nel caso di input testuali e variabili di uscita binarie.

Come per ogni modello di machine learning, è necessario, prima di utilizzare concretamente quest'ultimo, passare per la fase di addestramento.

Il modello è stato addestrato prendendo in input da HDFS un dataset costituito da tweet reali a cui è stato assegnato un valore numerico pari a zero nel caso di una valenza negativa, uno altrimenti. Sono necessarie, tuttavia, trasformazioni preliminari sull'input atte a rappresentarlo in un formato misurabile per il modello. Tali operazioni verranno descritte nel dettaglio nel capitolo successivo.

Dopodiché, il risultato finale della *pipeline* di trasformazioni viene "dato in pasto" al classificatore al fine di ottenere un modello addestrato.
Per garantire che il modello venga addestrato una volta soltanto e sia interrogato in modo efficiente, si è deciso di creare due script: uno relativo solo alla fase di training del modello e al suo salvataggio su HDFS e uno relativo soltanto all'utilizzo in produzione dello stesso, che permette all'utente l'interazione mediante riga di comando.

2. Dataflow e tecnologie utilizzate

Per poter realizzare il progetto, sono state utilizzate le seguenti tecnologie:

- **Apache Hadoop** per la "costruzione" del cluster e conseguentemente HDFS come tecnologia per il file system distribuito e YARN come *resource negotiator*;
- **Apache Spark e Spark MLlib** rispettivamente come tecnologia e libreria per la manipolazione dell'input e addestramento del modello di machine learning.

Nel dettaglio, il cluster è così configurato: sono presenti un nodo "Master", al cui interno sono in esecuzione i processi *Namenode, Secondary Namenode* (necessari per l'utilizzo di HDFS) ed il *Resource Manager* (necessario per YARN), e due nodi "Worker", che eseguono rispettivamente i processi *Datanode* (di HDFS) e *NodeManager* (di YARN).

Per i dettagli sulla configurazione, è possibile consultare il repository del progetto presente su github.

2.1 Fase di addestramento del classificatore

Per addestrare un modello, ottenendo buoni risultati, è necessario disporre di una grande quantità di dati (etichettati). È stato scelto il dataset di **Μαριος Μιχαηλιδης KazAnova** presente sulla piattaforma **Kaggle**, composto da 1,6 milioni di tweet (quindi è garantita una buona varianza dei dati) ognuno con i seguenti attributi:

- **target**: il sentimento of the tweet (0 = negativo, 1 = positivo);
- **ids**: id associato al tweet;
- **date**: la data di pubblicazione del tweet;
- **flag**: un flag riferito ad una manipolazione con Lyx;
- **user**: il proprietario del tweet;
- **text**: il contenuto del tweet.

Il flusso di lavoro è costituito dalle seguenti procedure:

1. i dati, disponibili in formato csv, vengono caricati all'interno di HDFS;
2. all'interno del programma, viene creato un **Dataset<Row>** a partire dai dati presenti su HDFS a cui vengono mantenute solo le informazioni utili all'addestramento, ovvero il contenuto dei tweet e il target associato.

Dataset<Row> è un oggetto che unisce i benefici degli RDD con i benefici di Spark SQL (come la possibilità di accedere alle singole "colonne" del dataset).

A questo punto, si procede con la manipolazione dei testi presenti.

3. si scompone ogni singolo testo in una lista di parole (***Tokenizzazione***);
4. vengono rimosse da ogni singola lista le congiunzioni, le preposizioni, gli articoli e tutte le cosiddette ***stop words***;
5. si costruisce, in funzione di tutti i testi presenti nel dataset di input, un ***bag of words***, un dizionario di tutte le parole rimanenti;
6. ogni lista di parole va trasformata in un vettore di numeri, in funzione dell'indice che ogni singola parola ha nel dizionario (questo comporta un passaggio da elementi di tipo string a elementi numerici);
7. si calcola, per ogni parola presente in ciascuna lista, l'***inverse document frequency***, una metrica che assegna ad ogni parola un'importanza inversamente proporzionale al numero delle occorrenze della stessa all'interno del dataset (più una parola è presente, meno informazione porta al classificatore);
8. viene eseguito il training del classificatore **Naive Bayes** e calcolata l'accuratezza delle predizioni su una porzione di dataset (il risultato viene salvato su HDFS)
9. viene salvato il modello su HDFS assieme ad alcuni modelli associati alle trasformazioni intermedie sopra descritte.

2.2 Fase di utilizzo del classificatore

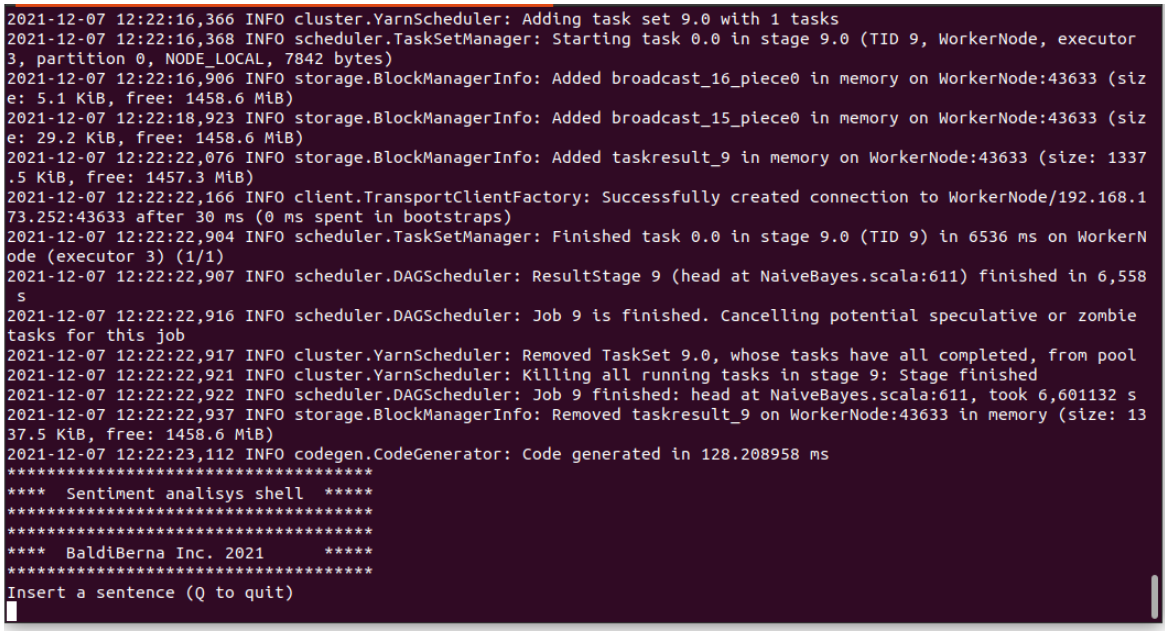
Il flusso è costituito da:

1. caricamento da HDFS dei modelli addestrati del classificatore e delle trasformazioni;
2. ricezione dell'input da riga di comando da parte dell'utente per mezzo di uno stream di caratteri;
3. trasformazione del testo mediante l'applicazione dei punti 3...7 presenti nel paragrafo 2.1;
4. stampa della classificazione calcolata dal modello sul terminale;

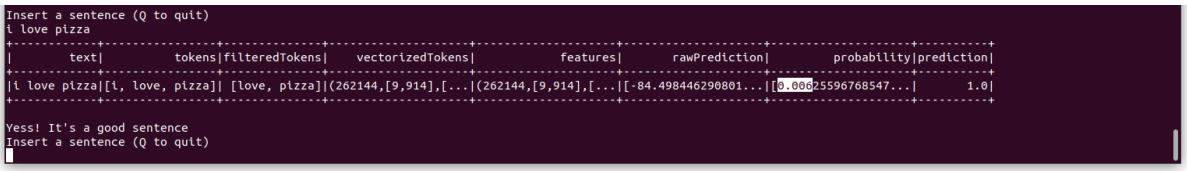
I punti 2,3 e 4 del flusso si ripetono financo l'utente non scrive il *carattere di escape*, che in questo caso è il carattere "Q".

3. Casi d'uso

Dopo che il modello è stato addestrato, l'utente può avviare il classificatore eseguendo, da riga di comando, lo script [consultabile su github]. A questo punto viene fatto il *submit* dell'applicazione spark ad hadoop e, dopo qualche minuto, all'utente comparirà sulla shell il seguente contenuto:



L'utente potrà quindi scrivere una frase, in inglese, sul terminale e otterrà la risposta, assieme a.



Questo scenario può essere iterato fino a quando l'utente non scriverà "Q" (mentre l'invio di una stringa vuota non porterà a nessuna eccezione); a quel punto l'applicazione verrà chiusa.

4. Limiti e possibili estensioni

Gran parte del tempo a disposizione è stato dedicato alla configurazione dei nodi per la creazione del cluster Hadoop, in quanto si è deciso di testare il sistema simulando un ambiente distribuito nel quale il nodo master e i nodi worker sono in esecuzione su macchine virtuali distinte e comunicanti attraverso la rete locale. A causa di ciò, non è stato possibile implementare o migliorare i seguenti aspetti:

- l'interazione utente-modello, che avviene per mezzo di una semplice riga di comando, potrebbe essere migliorata per mezzo della realizzazione di un'interfaccia grafica;
- i nodi del cluster potrebbero comunicare non solo tramite una rete locale ma anche attraverso la rete Internet;
- raffinare maggiormente l'accuratezza del modello (che attualmente si attesta al 75%) mediante tecniche di cross-validazione;
- estendere il supporto ad altre lingue oltre all'inglese.