Tesina di

Big Data Management

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

A.A. 2022/2023

docente
Fabrizio Montecchiani

Fake/Real News Prediction

350398 **Alessandra Vindice**

Indice

1. Introduzione

L'obiettivo di questo progetto è generare una predizione sulla veridicità di una notizia fornita da un utente attraverso un'applicazione web.
Il progetto si sviluppa in due fasi, la prima in cui viene trasformato il dataset e viene addestrato il modello di machine learning, la seconda in cui tale modello viene utilizzato.
Il modello scelto è il classificatore binario *Naive Bayes Classsifier*, il più utilizzato per l'analisi di input testuali. Esso viene creato attraverso un job da assegnare a Spark, sviluppato con Python tramite l'interfaccia PySpark. L'utilizzo di Spark integrato con Hadoop permette di eseguire il sistema in modo distribuito sia dal punto di vista del calcolo sia del file system.
In questa prima fase viene applicata all'input una pipeline di trasformazioni, opportunatamente salvata su HDFS, il cui risultato finale è passato al classificatore per ottenere il modello addestrato.
La seconda fase prevede lo sviluppo di un'applicazione web tramite il framework Flask, per consentire all'utente di inserire la notizia e ottenere la predizione. Questa applicazione, sviluppata in Python, permette di utilizzare il modello precedentemente creato, caricando quest'ultimo e la pipeline di trasformazioni da HDFS e applicando entrambi all'input.

2. Dataflow e tecnologie utilizzate

2.1 Dataset

Il dataset scelto per fornire i dati necessari all'addestramento del modello è il fake real news dataset, scaricato dalla piattaforma Kaggle.
I campi che lo compongono sono:

- **text**: la notizia;
- **label**: la veridicità della notizia (fake, real)

Il dataset è stato rielaborato per ottenere un unico csv e trasformare il campo label in binario (0 = fake, 1 = real). Per la memorizzazione del dataset è stato utilizzato il file system distribuito HDFS.

2.2 Tecnologie utilizzate

Per la realizzazione del progetto sono state utilizzate le seguenti tecnologie:

- **Apache Hadoop** con Hadoop Distributed File System (HDFS) come sistema di archiviazione distribuito e YARN come gestore delle risorse;
- **Apache Spark** per l'elaborazione dei dati sul cluster;
- **PySpark** come interfaccia Python per Spark;
- **MLlib** come libreria di machine learning per Spark;
- **Flask** come framework per lo sviluppo di applicazioni web in Python.

2.3 Dataflow

La prima fase del progetto permette di generare un modello di machine learning addestrato.
Attraverso l'esecuzione degli script .sh viene caricato il dataset su HDFS e viene lanciato Spark con il job scritto in Python. Il job, tramite le librerie di Pyspark, in particolare MLlib, esegue le seguenti operazioni in modo distribuito:

1. **Tokenizzazione**, ossia la scomposizione di ogni testo in una lista di parole;
2. Rimozione delle **stop words**;
3. Costruzione di un dizionario con tutte le parole presenti nei testi, **bag of words**;
4. Trasformazione della lista di parole presenti in un testo in un vettore di numeri che dipende dall'indice che ogni parola ha nel dizionario;
5. Calcolo dell'inverse document frequency (**IDF**), una metrica che assegna ad ogni parola un'importanza inversamente proporzionale al numero di occorrenze della stessa all'interno del dataset;
6. Esecuzione dell'addestramento del classificatore Naive Bayes e calcolo dell'accuracy;
7. Salvataggio delle trasformazioni e del modello su HDFS.

L'accuracy del modello è 0.92 quindi risulta essere soddisfacente.
Successivamente c'è la fase di utilizzo del modello attraverso un'applicazione web Flask.
Il framework Flask è utilizzato per costruire l'applicazione e gestire la logica lato server. L'interfaccia utente viene creata con HTML e Javascript e viene utilizzata per inserire la notizia da processare. Quando l'input viene inviato, viene mandata una richiesta AJAX al server con il dato inserito.
Lato server verrà eseguito uno script Python che processa il dato di input passandolo come argomento al codice Pyspark. Il job carica i modelli da HDFS ed esegue la pipeline di trasformazioni dell'input e il calcolo della predizione.
Una volta completato il job, l'output viene restituito all'applicazione web usando una risposta AJAX. Infine, la predizione relativa alla notizia inserita viene visualizzata sull'interfaccia utente.

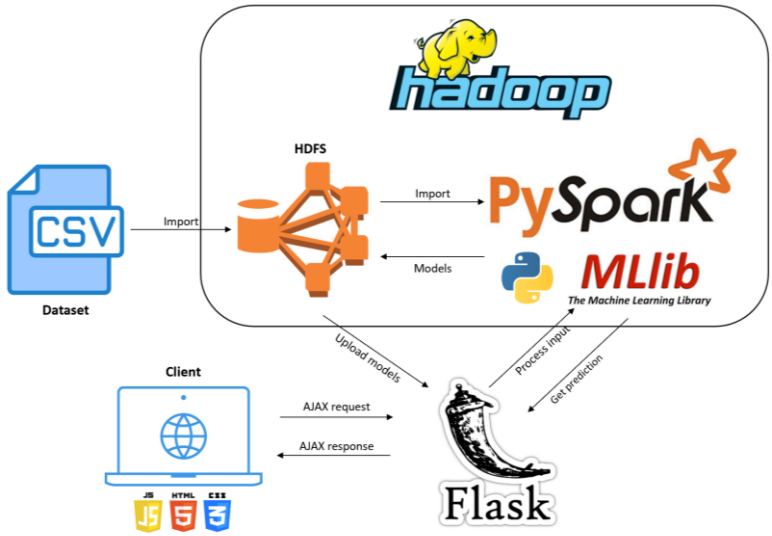

Figura 1: Architettura del sistema

3. Casi d'uso

Per eseguire l'addestramento del modello è stato creato lo script runModel.sh che automatizza l'attivazione di Hadoop, l'esecuzione del job Spark e il salvataggio dei modelli in HDFS.
Per prima cosa va impostato dentro lo script runSpark.sh la variabile MY_PTH specificando il percorso per arrivare alla cartella PySpark.
Successivamente bisogna lanciare lo script:
sh runModel.sh
Una volta completato l'addestramento del modello, si lancia lo script runApp.sh che esegue il comando flask run. In questo modo viene avviato il server che, all'indirizzo http://localhost:5000, mostra l'interfaccia web dell'applicazione, dove è possibile inserire la notizia in inglese e predire la sua veridicità.

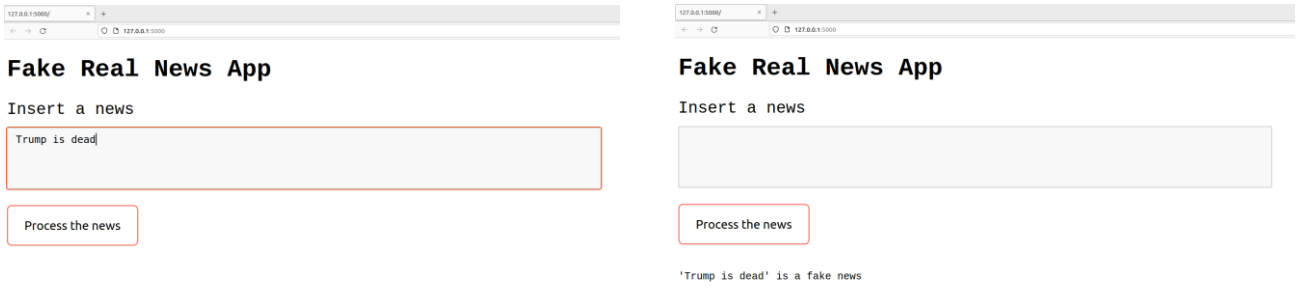

Figura 2: Interfaccia utente

4. Limiti e possibili estensioni

Questo sistema ha sicuramente delle limitazioni e potrebbero essere apportate delle migliorie per superarle e aumentare le funzionalità.
Il limite principale è che è possibile processare solo notizie in inglese, quindi il sistema potrebbe essere esteso per supportare più lingue.
Inoltre, potrebbe essere migliorata l'interfaccia grafica e potrebbe essere implementata un'interfaccia per i dispositivi mobili.